AEC

Advances in Enterprise Control

Symposium Sponsored by JFACC Program, DARPA-ISO

AEC Proceedings

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
10-07-02000

2. REPORT TYPE
Symposium Proceedings

3. DATES COVERED *(From - To)*
N/A

4. TITLE AND SUBTITLE
Advances in Enterprise Control Symposium Proceedings

5a. CONTRACT NUMBER
N/A

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
N/A

6. AUTHOR(S)
Multiple (See Document)

5d. PROJECT NUMBER
N/A

5e. TASK NUMBER
N/A

5f. WORK UNIT NUMBER
N/A

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Multiple (See Document)

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
DARPA/ISO
JFACC Program
3701 N. Fairfax Drive
Arlington VA 22203-1714

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
N/A

12. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution Statement A: Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The 2000 DARPA-JFACC Symposium on Advances in Enterprise Control was organized under the sponsorship of the Joint Force Air Component Commander (JFACC) Program in the Information Systems Office (ISO) of the Defense Advanced Research Projects Agency (DARPA). The purpose of this symposium was to bring together researchers and practitioners from industry, Government and academia to present and discuss the latest developments in all aspects of enterprise control. The participants of the Symposium presented papers that (a) described the results of original research on enterprise control, (b) provided broad reviews of the state-of-the-art theory and techniques, and (c) proposed and advocated new research directions.

15. SUBJECT TERMS Enterprise control, agent-based systems, adversarial games, model-predictive control, hierarchical models, Markov decision processes, roling-horizon optimization, dynamics programming

16. SECURITY CLASSIFICATION OF: **UNCLASSIFIED**			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Jackie E. Lynch
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	None	240	**19b. TELEPHONE NUMBER** *(include area code)* 703-526-6609

Symposium
July 2000
Minneapolis MN

AEC

Advances in Enterprise Control

Advances in Enterprise Control
Symposium Proceedings

Sponsored by:

JFACC Program
Defense Advanced Research Projects Agency (DARPA)
Information Systems Office (ISO)

Minneapolis, Minnesota
July 10-11, 2000

TABLE OF CONTENTS

Section 3: Adversarial Games: Models & Solutions

Section 4: Variable Granularity Models: Abstractions & Decompositions

Preface

The 2000 DARPA-JFACC Symposium on Advances in Enterprise Control was organized under the sponsorship of the Joint Force Air Component Commander (JFACC) Program in the Information Systems Office (ISO) of the Defense Advanced Research Projects Agency (DARPA). The purpose of this symposium was to bring together researchers and practitioners from industry, government and academia to present and discuss the latest developments in all aspects of enterprise control. The participants of the Symposium presented papers the (a) described the results of original research on enterprise control, (b) provided broad reviews of the state-of-the-art theory and techniques, and (c) proposed and advocated new research directions.

The modern enterprise is a large-scale dynamic system with broadly distributed and potentially conflicting goals, resources and constraints, with multiple semi-autonomous participants of both human and artificial nature (e.g., large military operations, financial/trading institutions, logistics systems, manufacturing plants, power grids., etc). The increasing capabilities of technology to collect, automatically generate, and disseminate information offer the possibility for large-scale enterprises to be more responsive to change. Enterprise plans and orders quickly become obsolete as new information about the current situation becomes available. The challenge is to use real-time information to re-direct enterprise operations effectively. Such systems and challenges defined the scope of the Symposium.

PROGRAM COMMITTEE

Dr. Richard Brooks, Pennsylvania State University
rrb5@psu.edu

Dr. David A. Castanon, Boston University
dac@bu.edu

Dr. Michael C. Caramanis, Boston University
mcc@enga.bu.edu

Dr. Datta Godbole, Honeywell Technology Center
godbole_data@htc.honeywell.com

Dr. Alexander Kott, Logica Carnegie Group
kotta@logica.com

Dr. Bruce H. Krogh, Carnegie Mellon University
krogh@ece.cmu.edu

Dr. Nelson D. Ludlow, Orincon Corporation
nludlow@orincon.com

Dr. Peter B. Luh, University of Connecticut
luh@engr.uconn.edu

Sr. Marwan A. Simaan, University of Pittsburgh
simaan@pitt.edu

Dr. Boris Stilman, University of Colorado at Denver
bstilman@cse.cudenver.edu

FOREWORD

There is a revolution underway in the management of large enterprises. In applications ranging from manufacturing supply chains to military operations, conventional methods and tools have become outmoded because information technology now makes it possible to become aware of new developments (i.e., disturbances, results, information) almost instantaneously. The increasing capabilities of technology to collect, automatically generate, and disseminate information offer the possibility for large-scale enterprises to be more responsive to change. But this means enterprise plans and orders quickly become obsolete as new information about the current situation becomes available. The problem is no longer how to acquire information, but rather what to do with the flood of real-time data that makes any plan obsolete almost before it can be computed.

Recognition of this situation led the Joint Force Air Component Commander (JFACC) Program in the Information Systems Office (ISO) of the Defense Advanced Research Projects Agency (DARPA) to sponsor the 1st Symposium on Advances in Enterprise Control (AEC) in November 1999 in San Diego CA. Of interest were all modern organizations that can be characterized as large-scale dynamic systems with broadly distributed and potentially conflicting goals, resources and constraints, and with multiple semi-autonomous participants -- both human and artificial. Examples include large military operations, financial/trading institutions, logistics systems, manufacturing plants, and power grids. The purpose was to bring together researchers from academe, industry and government to present and exchange new ideas on how to use real-time information to re-direct and control such enterprises effectively.

This volume is the Proceedings of the 2nd AEC Symposium, held July 10-11, 2000 in Minneapolis, MN. As with the first symposium, the meeting in Minneapolis included papers on a wide variety of applications, techniques and approaches reflecting the breadth and complexity of the problems faced by contemporary enterprises. Three invited talks provided insight into the common problems shared across seemingly disparate domains. Dr. Massoud Amin of EPRI presented the research being conducted in the ERPI-DoD Complex Interactive Networks/Systems Initiative, focusing on the US power grid and other networks critical to the national infrastructure. Dr. Joseph Knickmeyer of the Military Traffic Management Command Transportation Engineering Agency spoke on the problems of logistics planning and execution to support U.S. military operations worldwide. Professor Sridhar Tayur of the Graduate School of Industrial Administration at Carnegie Mellon University described new developments in the internet-enabled management of supply chains. Each of these talks highlighted the need for new paradigms for understanding and designing enterprise control systems.

This volume contains the contributed papers presented at the 2nd AEC Symposium. The presentations were organized into four sessions, each addressing principal themes in current research on enterprise control systems. The papers in the first session, titled "Distributed Control & Agent-Based Systems," emphasized that with large enterprises, control necessarily becomes decentralized and distributed. Decision-makers typically exhibit some independence in operating objectives and capabilities; that is, they act more like agents than simple automata. Papers in this session included techniques for modeling and evaluating complex agent-based systems. The design and impact of the enterprise organization was also addressed.

In many cases, the architecture and technology for information exchange and communication protocols can be as important to the effectiveness of an enterprise control system as the operating constraints and availability of resources. Papers in the second session, titled "Information Flow and Exchange," explored these issues. One study demonstrated how seemingly benign and standard methods for dealing with communication errors could lead to disastrous results when the dynamics of enterprise control systems are taken into account. Another paper dealt with collaboration in a distributed enterprise. New technologies for sensing and network-based dissemination of information were also presented.

FOREWORD

With the existence of multiple decision-makers both inside and outside of an enterprise, some of whom have conflicting and even hostile intents, there is a renewed interest in game theory and its application to real-world problems. We dedicated a separate session, titled "Adversarial Games: Models and Solutions," to these issues. Within the broad outlines of this theme, the authors explored the application of Dynamic Programming to adaptive scheduling in a risky environment; construction of a game theoretical model involving resource allocation; issues of deception and dealing with partial information; and uses of model-predictive control in hostile environments.

Finally, all of the papers, in some way, dealt with enterprise modeling, either implicitly or explicitly. (Indeed, some researchers believe modeling is *the* issue in enterprise control.) The final session, titled "Variable Granularity Models: Abstractions and Decompositions," focused on how enterprise models can be exploited to make fundamental problems of analysis and design tractable. Themes included the issue of consistency in hierarchical models, the use of decomposition for computing strategies in a stochastic optimization formulation, and the use of reduced finite-state models to avoid the "curse of dimensionality" in computing solutions to dynamic games. The use of formal methods from computer science to verify properties of large enterprise models was also presented.

Throughout these papers, there are many references to concepts from classical control and decision theory, such as stability, Markov decision processes, rolling-horizon optimization, and dynamic programming. This is not surprising since the availability of real-time feedback is the principal impetus for new research in enterprise control systems. But this is not to say the problems are solved. If there is any one conclusion that emerges from these papers it is that *enterprise* control is not a standard control problem at all. It is not simply a matter of applying known techniques and results in a new context. Indeed, enterprise control requires new approaches and begs for solutions to many of the largely unsolved problems in the control theory literature. Some old themes are being taken up again, not because the problems have been solved already, but rather because there is a new reason to try and make progress on unsolved problems that have long been recognized as extremely difficult and challenging.

We want to thank all of the authors for their contributions to the 2nd AEC Symposium and to this volume. We hope it will be a valuable reference for researchers currently working on problems in enterprise control, and that it will attract more researchers to this exciting field. As these papers demonstrate, advances are being made, but there is plenty of work yet to be done.

Alex Kott
Logica Carnegie Group, Inc.

Bruce H. Krogh
Carnegie Mellon University

Invited Talk 1

EPR/DoD Complex Interactivc Networks/Systems Initative: Self-Healing Infrastructures

EPRI/DoD Complex Interactive Networks/Systems Initiative: Self-Healing Infrastructures[1]

Massoud Amin, D.Sc.
Electric Power Research Institute (EPRI)
3412 Hillview Ave.
Palo Alto, CA 94304-1395, USA
e-mail: mamin@epri.com; phone: 650.855.2452

ABSTRACT

Energy, telecommunications, transportation, and financial infrastructures are becoming increasingly interconnected, thus, posing new challenges for their secure, reliable and efficient operation. All of these infrastructures are, themselves, complex networks, geographically dispersed, non-linear, and interacting both among themselves and with their human owners, operators, and users. No single entity has complete control of these multi-scale, distributed, highly interactive networks, nor does any such entity have the ability to evaluate, monitor, and manage them in real time. In fact, the conventional mathematical methodologies that underpin today's modeling, simulation, and control paradigms are unable to handle the complexity and interconnectedness of these critical infrastructures.

There is reasonable concern that national and international, energy and information infrastructures have reached a level of complexity and interconnection which makes them particularly vulnerable to cascading outages, initiated by material failure, natural calamities, intentional attack, or human error. Secure and reliable operation of these networks is fundamental to national and international economy, security and quality of life.

In a joint initiative with the Deputy Under Secretary of Defense for Science and Technology, through the Army Research Office (ARO), EPRI is working to develop new tools and techniques that enable large national infrastructures to function in ways that are self-healing. The Complex Interactive Networks/ Systems Initiative (CIN/SI) is a 5-year, $30 million program of Government Industry Collaborative University Research (GICUR), funded equally by DoD and EPRI. This paper provides a brief summary of CIN/SI.

INTRODUCTION

As the complexity of national infrastructures and their intertwined operations have increased, so have the human benefits they provide; these continental-scale networks have become responsible for much of "the good life" that, at least in the more developed countries, is led today. However, with those increasing benefits come increasing risks. Local actions have the potential to create global effects by cascading throughout a network and even into other networks, making them vulnerable to failures with widespread consequences. Interactions between these individual networks increase the complexity of their operation and control.

As an example, a growing portion of the world's business and industry, art and science, entertainment and even crime are conducted through the World Wide Web and the Internet. But the use of these electronic information systems depends, as do the more mundane activities of daily life, on many other complex infrastructures, such as cable and wireless telecommunications, banking and finance, gas, water and oil pipelines, the electric power grid, and transportation. These interactive networked systems present unique challenges for robust control and reliable operation, such as:

- Multi-scale, heterogeneous, multi-component, and distributed nature of these large-scale interconnected systems;
- Vulnerable to attacks and local disturbances which can lead to widespread failure almost instantaneously;
- Characterized by many points of interaction among a variety of participants – owners, operators, sellers, buyers, customers, data and information providers, data and information users;
- The number of possible interactions increases dramatically as the number of participants grows. As a result, the complex activity of these networks greatly exceeds the ability of a single centralized entity to evaluate, monitor, and manage them in real time; and
- Too complex for conventional mathematical theories and control methodologies.

[1] Keynote presentation at the 2nd DARPA-JFACC Symp. on Advances in Enterprise Control, Minneapolis, July 10-11, 2000

The North American power network, for example, may realistically be considered to be the largest machine in the world—its transmission lines connect all the electric generation and distribution on the continent. But this network was developed over the last 100 years without a conscious awareness and analysis of the system-wide implications of its current evolution under the forces of deregulation, the digital economy, and interaction with other infrastructures. Only recently has the possibility of power delivery beyond neighboring areas become a key design and engineering consideration, yet the existing grid is being required to handle a growing volume and variety of long-distance, bulk power transfers. Grid congestion and atypical power flows are increasing, while customer expectations of reliability are rising to meet the needs of a pervasively digital world.

Widespread outages and huge price spikes during the last 4 years have raised public concern about grid reliability at the national level. The potential for larger-scale and more frequent power disruptions is considered higher now than at any time since the great Northeast blackout of 1965. Furthermore, the potential ramifications of network failures have never been greater, as the transportation, telecommunications, oil and gas, banking and finance, and other infrastructures depend on the continental power grid to energize and control their operations.

From a broader view, the various areas of interactive infrastructure networks present numerous theoretical and practical challenges in modeling, prediction, simulation, cause and effect relationships, analysis, optimization and control of coupled systems comprised of a heterogeneous mixture of dynamic, interactive, and often nonlinear entities, unscheduled discontinuities, and numerous other significant effects. In many complex networks, for instance in the organization of a corporation, the human participants are both the most susceptible to failure and the most adaptable in the management of recovery. Modeling these networks, especially in the case of economic and financial market simulations will require modeling the bounded rationality of actual human thinking, unlike that of a hypothetical "expert" human as in most applications of artificial intelligence. Furthermore, a pertinent question is at what resolution should sensing, modeling, and control be started to achieve the overall objectives of efficiency, robustness and reliability?

Secure and reliable operation of these systems is fundamental to national and international economy, security and quality of life. Their very interconnectedness makes them more vulnerable to global disruption, initiated locally by material failure, natural calamities, intentional attack, or human error. Because a change in conditions at any one location can have immediate impacts over a wide area, a local disturbance's effects can be magnified as they propagate through a network. Cascading failures can occur—almost instantaneously—with major consequences in geographically remote regions or seemingly unrelated businesses.

A reasonable concern for these risks is well warranted. In the United States, the President's Commission on Critical Infrastructure Protection issued a report in October 1997 that stressed the need for research to enhance the security of complex interactive infrastructure networks. The report cited their growing importance in many application areas, and the potentially damaging and even dangerous economic, security and health impacts of the undesirable propagation of disturbances throughout them.

Since humans interact with these infrastructures as managers, operators and users, human performance plays an important role in their efficiency and security. After briefly describing the overall CIN/SI program objectives, this paper provides details on the specific areas of work and some of the issues that arise in these complex networks. Pertinent to this symposium are the areas of enterprise modeling and human performance in critical infrastructures; for more information on modeling the human factors in this context, please see (Wildberger and Amin 2000)

CIN/SI: EPRI/DOD COMPLEX INTERACTIVE NETWORKS/SYSTEMS INITIATIVE

There are clearly many opportunities for modeling and simulation, as well as the employment of machine intelligence and human factors engineering in this area. Mathematical models of such complex systems are typically vague (or may not even exist); moreover, existing and classical methods of solution are either not available, or are not sufficiently powerful. Management of disturbances in all such networks, and prevention of undesirable cascading effects throughout and between networks, requires a basic understanding of true system dynamics, rather than mere sequences of steady-state operations. Effective, intelligent, distributed control is required that would enable parts of the networks to remain operational and even automatically re-configure in the event of local failures or threats of failure.

The Complex Interactive Networks/Systems Initiative (CIN/SI) is a 5-year, $30 million program of Government Industry Collaborative University Research (GICUR), funded equally by EPRI (http://www.epri.com/targetST.asp?program=83) and the United States Department of Defense (DoD), through the Army Research Office (ARO). (http://www.aro.army.mil/research/complex.htm) The objective of this initiative is to produce a significant, strategic advancement in the robustness, reliability, and efficiency of the interdependent energy, communications, financial, and transportation infrastructures.

A key concern is the avoidance of widespread network failure due to cascading and interactive effects. Of course, DoD is more concerned with intentional disturbances by an enemy, while EPRI is more interested in natural disasters and material failures. Although forecasting their likelihood and expected location may require different skills, there is very little difference in the effects and the task of recovery whether the power pole or the telephone switch box was destroyed by lightning or by terrorist attack.

CIN/SI was initiated in mid-1998, and work began in spring 1999. Through a highly competitive source selection process, CIN/SI has funded six consortia, consisting of 28 universities. Commonwealth Edison Co. and Tennessee Valley Authority are also participating directly in the program, providing staff expertise, data, test and demonstration sites for innovative modeling, measurement, control, and management tools. CIN/SI was launched to develop:

- Methodologies for robust distributed control of heterogeneous, dynamic, widely dispersed, yet interconnected systems;
- Techniques for exploring interactive networked systems at the micro- and macro-levels; and
- Tools to prevent and/or ameliorate cascading effects through and between networks.

Work focuses on advancing basic knowledge and developing breakthrough concepts in modeling and simulation; measurement, sensing, and visualization; control systems; and operations and management. In order to achieve this goal, technical objectives were defined in three broad areas:

- **Modeling:** Understanding the "true" dynamics—to develop techniques and simulation tools that help build a basic understanding of the dynamics of complex infrastructures.
- **Measurement**: Knowing what is or will be happening—to develop measurement techniques for visualizing and analyzing large-scale emergent behavior in complex infrastructures.
- **Management**: Deciding what to do—to develop distributed systems of management and control to keep infrastructures robust and operational.

Specific needs in each of these areas are enumerated below:

Modeling

Qualitative and quantitative models of complex interactive systems, including

- Formal methods for modeling true dynamics and for real-time computation to cope with system uncertainties and establish provable performance bounds;
- Multi-resolution simulations, with the ability to go from the macro- to the micro- level, and vice versa;
- "Artificial life" (cellular automata and multi-agent models) for modeling and solving otherwise intractable problems in networked systems;
- Optimization and control theory along with decision analysis to model hybrid (mixed discrete/continuous) systems; and
- Techniques for on-line mathematical modeling and decision support with partial inputs and in the presence of errors.

Measurement

Analytical and computational tools for measuring large-scale complex networks, including

- Real-time survey and status monitoring of systems;
- Real-time processing of large data sets via pattern extraction (data mining and cluster analysis) and other means;
- Techniques for correlating information from separate data sources/sensors;
- Intelligent sensors and actuators;
- Tools and techniques for system verification and validation;
- Adaptive strategies that help components discern their interactions with the environment; and
- Methods for providing feedback about key environmental variables and for generating appropriate commands using local computational devices.

Management

A comprehensive framework for distributed network management, including

- Real-time system state analysis;
- Open architectures, intelligent devices, and distributed multi-level controllers;
- Methods for reasoning, planning, negotiation, and optimization;
- Methods for rule generation and modification;
- Automatic verification of real-time, adaptive systems using formal proofs from specifications;
- Task coordination of multiple intelligent agents (both artificial and human) in uncertain dynamic systems;
- Tools for automated negotiation and risk management among self-interested agents (e.g., game theory with computational and resource bounds);
- Algorithms for "optimal" performance by independent agents with independent objectives;
- Overall control techniques in environments where intelligent response devices may be acting against each other;
- Methods for accommodating structural uncertainty and limiting impacts of system disturbances;
- Methods for predicting impending failures: root-cause modeling for real-time diagnosis; early warning and failure forecasting; and
- Methods for recovering from emergencies.

DETAILS ON SPECIFIC RESEARCH AREAS:

Robust Control

Design of self-healing systems requires the extension of the theory of robust control in several ways beyond its present focus on the relatively narrow problem of feedback control:

- Robust bifurcation analysis: understanding relationships between bifurcation and cascading events and finding the relevant low-order manifolds where bifurcation is taking place, while tolerating errors and uncertainty in the remaining components of the system; in the case of power systems, dynamical analysis requires both nonlinear swing-equation models for frequency/active power and new models for voltage/reactive power dynamics
- Prescriptive approaches to prevention of cascading network failures
- Trade-off analyses regarding sensitivity to cascading events in designed systems
- Conservation laws on the achievable robustness of interconnected systems
- Parameter robustness tools for analysis and design of power system stabilizers
- Prediction and detection of the onset of failures both at the local and global network levels, followed by the generation of actions to prevent the propagation of disturbances
- Adaptive nonlinear algorithms to control network performance via fault tolerance, re-routing, redundancy allocation, and maintenance actions
- Distributed control: establishing the appropriate degree of centralization, establishing communication/information requirements, and providing multi-objective evaluation; context-dependent agent coordination and learning; distributed hybrid control of multi-agent networked systems

Disturbance Propagation in Networks

Prediction and detection of the onset of failures both in local and global network levels:

- Fault diagnosis and correction
- Realistic and robust models of random maps from faults to sequences of alarms
- Algorithms for identification of faults that cause the observed alarms
- Stochastic characterization of cascading, failure models, and interaction
- Inter-/intranetwork causality and dependence
- Parsimonious stochastic models of failure

Complex Systems

Theoretical underpinnings of complex interactive systems:

- Robustness analysis in complex distributed systems, based on implicit rather than input-output modeling, and on the *gap metric* between interconnected components as a low-complexity method to measure global sensitivity
- Model reduction in distributed systems; emphasis on nonlinear differential-algebraic models
- Phase transitions and statistical physics for engineered complex systems, specifically with regard to critical behavior and analysis of phase transitions in network models
- Emergence of collective phenomena: how can emergent phenomena be detected early, and how can they be controlled?

Dynamic Interaction in Interdependent Networks

Characterization of uncertainty in large distributed and layered networks:

- Multi-resolution techniques where various levels of aggregation can co-exist
- Use of system invariances as a technique for complexity reduction in design with global specifications
- Power systems as layered networks
- The physical power system

- Sensors interconnected by a communication network
- Sensors controlled by a computer network
- Economic signals from agents linked through economic markets and auctions.
- Mathematical strategies for model reduction of complex dynamical systems defined by layered networks
- Formulation of admission control and flow regulation strategies in networks
- Game-theoretic and team theoretic concepts
- Imperfect information, delays, and multiple time scales
- Software tools for the synthesis and verification of safety-critical hybrid control systems for complex networks both in the normal mode of operation and in degraded modes of operation

Modeling in General

- Generic research and idealized models, consisting of static graph-theoretic models; and interactive dynamic models, such as interconnected differential-algebraic systems
- Internet models, describing network-level phenomena
- Modeling issues under hard constraints (less than a second time constraints especially with regard to transient stability)
- Hybrid models
- Time-stepped fluid simulation of communication networks
- Controlling time-driven dynamics by controlling specific discrete events
- Discrete event dynamic system models: performance evaluation and scheduling; development of networked agent theory and incentive-based control theory in networks; meeting global efficiency goals and local security objectives
- Power system models
- Swing models of power networks, based on linearized and nonlinear swing dynamics, and assuming no voltage dynamics.
- Coupled voltage-swing models, taking into account voltage and reactive-power dynamics as well as swing dynamics, but still small scale
- Modeling of power electronics (SMES, SVC, TCSC, UPFC, PSS), and other local control equipment

Forecasting; Handling Uncertainty and Risk

- Characterizing uncertainties and managing risk: identifying those uncertainties in the operating conditions that affect the security level of the power system
- Candidates are the total demand, the allocation of the demand among the buses, the power factor at each bus, and generation and voltage setpoints at each bus
- Hierarchical and multi-resolution modeling and identification in power systems
- Uncertainty characterizations describing the different model resolutions
- Robustness analysis of relevant properties such as system bifurcations
- Stochastic analysis of network performance
- Paradigms for uncertainty and its propagation, i.e., probability theory or belief functions
- Ordinal optimization: predicting load and market clearing prices; taking other parties' decisions into consideration when deciding one's own bids; risk and bounded rationality
- Handling rare events: large deviations theory for identifying and detecting mechanisms of rare event failures in power system dynamics coupled with stochastic discrete event models
- Identifying complex mechanisms of cascading failures, and detecting their associated "signatures" in system measurements
- Modeling framework and analytical tools for studying the dynamics and failure modes in the interaction between economic markets and power systems via a delay- and loss-prone communication network
- Possible models will include piece-wise deterministic Markov processes
- Analytical tools may involve (a) limit theorems under different scalings that result in model simplification, (b) stability analysis, and (c) identification of failure modes via the existence of multiple equilibria in deterministic dynamical system limits
- Market instability failure mechanisms and market thresholds for identifying their onset
- Considering modes of failure that may occur as a result of interactions between markets and power systems
- Models include components of stochastic effects, hybrid representations, multiple time scales, nonlinear effects (including human factors effects), time variation effects, weather effects, and effects brought about by changes in the economic infrastructure itself
- Efficient simulation techniques for failure events in power system networks
- Application of quick simulation techniques and importance sampling for hidden failure location and risk evaluation of major cascading disturbances and blackouts

- Modeling the failure event as the sequence of states taken on by a Markov chain

Areas of research being investigated in CIN/SI by each consortium, their foci and solution components are depicted in Figures 1-2:

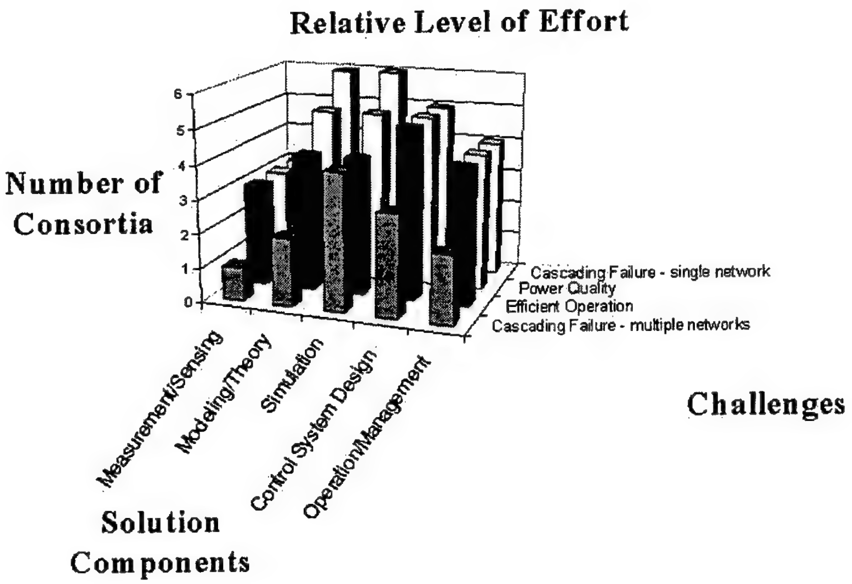


Figure 1: Relative Level of Effort vs. Challenges and Solution Components

Operation Efficiency and Effectiveness	**Harvard-4 Purdue-2 U WA-2**	**Caltech-1 Cornell-1, 2 Harvard-1, 3 Purdue-1.3, 3.1, 3.4**	**CMU-2, 3 Harvard-1 Purdue-3.2, 3.3 U WA-4.2,4.3, 5.3, 5.4**	**Caltech-1, 2, 4 CMU-4 Cornell-1, 3 Harvard-3 U WA-1.3, 4.4,5.1**	**CMU-5, 7 Harvard-3 Purdue-4,5 U WA-3**
Robustness and Security	**Harvard-4 Purdue-2 U WA-2**	**Caltech-1 CMU-1, 2 Cornell-1, 2 Harvard-1, 3 Purdue-1.3, 3.1, 3.4**	**Caltech-3 CMU-3, 5 Harvard-1 Purdue-3.2, 3.3 UWA-4.2-3, 5.3-4**	**Caltech-1, 2, 4 CMU-3, 4 Cornell-1, 3 Harvard-3 U WA-4.4,5.1**	**CMU-7 Harvard-3 Purdue-4,5 U WA-3**
Cascading Failure (single network)	**Harvard-4 Purdue-1.2, 2 U WA-1.4, 2, 4.5**	**Caltech-1, 2 CMU-2 Cornell-4.2, 7 Harvard-1, 2, 3 Purdue-1.3, 3.1, 3.4 U WA-1.1**	**Caltech-3 CMU-3, 5 Cornell-6 Harvard-1, 2 Purdue-3.2, 3.3 UWA-4.2-3, 5.3-4**	**Caltech-1, 2, 4 CMU-3, 4 Cornell-3, 5, 6 Harvard-2, 3 UWA-1.3, 4.4,5.1**	**CMU-7 Cornell-4.1, 5 Purdue-4,5 U WA-3**
Cascading Failure (multiple networks)	**Harvard-4**	**Caltech-5 Cornell-7**	**Caltech-3, 5 Cornell-4.3 Harvard-3 U WA-5.5**	**Cornell-3 Harvard-3 U WA-5.5**	**Cornell-4.4 U WA-5.5**
	Measurement, Sensing and Visualization	**Modeling/ Theory**	**Simulation**	**Control Systems Design**	**Operation and Management**

Solution Component

Figure 2: A canvas of research and development for reliable and robust operation vs. solution components
Consortia Lead Universities: Caltech; Carnegie Mellon U; Cornell U; Harvard U; Purdue U; U of Washington
Notation: Lead University for each consortium is noted along with the particular task number in the SOW

MODELING AND HUMAN PERFORMANCE

National and international infrastructures are characterized by many points of interaction among a variety of human participants—owners, operators, and maintenance personnel, as well as users of all kinds. In many complex networks, the human participants themselves are both the most susceptible to failure and the most adaptable in the management of recovery. Modeling and simulating these networks, especially their economic and financial aspects, will require modeling the bounded rationality of actual human thinking, unlike that of a hypothetical "expert" human as in most applications of artificial intelligence (AI). Even more directly, most of these networks require some human intervention for their routine control and especially when they are exhibiting anomalous behavior that may suggest actual or incipient failure.

Operators and maintenance personnel are obviously “inside” these networks and can have direct, real-time effects on them. But the users of a telecommunication, transportation, electric power or pipeline system also affect the behavior of those systems, often without conscious intent. The amounts, and often the nature, of the demands put on the network can be the immediate cause of conflict, diminished performance and even collapse. Reflected harmonics from one user’s machinery degrade power quality for all. Long transmissions from a few users create Internet congestion. Simultaneous lawn watering drops the water pressure for everyone. In a very real sense, no one is “outside” the infrastructure.

Given that there is some automatic way to detect actual or immanent local failures, the obvious next step is to warn the operators. Unfortunately, the operators are usually busy with other tasks, sometimes even responding to previous warnings. In the worst case, the detected failure sets off a multitude of almost simultaneous alarms as it begins to cascade through the system, and, before the operators can determine the real source of the problem, the whole network has shut itself down automatically.

Unfortunately, humans have cognitive limitations that can cause them to make serious mistakes when they are interrupted. In recent years, a number of systems have been designed that allow users to delegate tasks to intelligent software assistants (“softbots”) that operate in the background, handling routine tasks and informing the operators in accordance with some protocol that establishes the level of their delegated authority to act independently. In this arrangement, the operator becomes a supervisor, who must either cede almost all authority to subordinates or be subject to interruption by them.

At present, we have very limited understanding of how to design user interfaces to accommodate interruption. Among other studies, the Human Alerting and Interruption Logistics (HAIL) Project at the Naval Research Laboratory is currently addressing this issue. (McFarlane 1997, 1999)

Complexity and Automation

The most common response when faced with a complex task is to try to engineer the human out of the loop by automating all or part of the task. Ever increasing complexity and automation are interrelated secular trends affecting both the operation and the management of technology in ways that have particularly significant impacts on human performance. Increased automation changes, rather than reduces, the importance of human performance. In most cases, the human role is shifted from online, active, operational control to supervisory control, maintenance of the automated equipment, and design of control strategies. The growing capability and adaptability of the human/machine system increases the overall complexity that must then be managed either by the human or by yet more automation. If the human is required to manage this additional complexity, the workload increases and the required skills become more difficult to find or to train. (Dunn and McBride 1999) So the usual next step is to further increase the automation. This response makes the system being managed ever more distant from its human managers. Their lack of any detailed understanding then makes the whole process more unpredictable. (Tenner 1997) It is easy to see why there is a widespread sense that automation seems to trade the reduction of physical labor for a corresponding increase in mental stress.

Users can “make or break” any system, and it is the responsibility of the designers of the system to make it as “user-proof” as possible. However, most of the infrastructure networks were not designed at all. They simply grew. The Internet is the extreme example of this phenomenon, but neither the telephone nor the electric power networks were specifically designed for the kind of use they are now getting, much less the kind they can realistically expect in the near future. Electric power is now being routinely traded over interconnections that were originally installed only for emergency back-up. Telephone line capacities and service capabilities vary geographically, sometimes even within the same city or county. Of course, users adapt their use of the infrastructure based on the response they get from it. But their individual adaptation may be counter-productive even to their own needs, much less to the general benefit of all users.

There are two basic ways to influence usage behavior: designed incentives and open information. If the particular network is managed, and especially if usage is fee based, incentives can be arranged to spread usage/traffic. Time differential long distance phone rates are one example of this and demand side management of electric power with contractual rebates is another. (EPRI 1993) Ideally, these differentials would vary with demand in real-time. Modern instrumentation and computer capability is beginning to make this possible, but we still need better models of human response to these incentives. Individual differences, based on historical billing data can now be included in the models. Forecasting future demand will also have to take "gaming" tactics into account. (Friedman and Rust 1993)

There is considerable evidence that, given enough information, humans are willing to cooperate for their mutual benefit. (Friedman and Rust 1993) Even networks that are effectively a "free good," like the Internet, can achieve more efficient use, and avoid some forms of failure, by providing all users with the appropriate information on which to base their usage decisions. New fractal models of message traffic (Willinger and Paxson 1998) and strong empirical evidence of Pareto (1927) optimization in search behavior on the World Wide Web may provide insights on how to maximize benefits of this and similar electronic communications for all their users. (http://www.parc.xerox.com/spl/groups/dynamics/www/new.html)

Reducing the Management Knowledge Deficit
Managers of technically complex systems like the national infrastructure networks seldom have a detailed knowledge of underlying technologies. It is arguable as to whether or not they need such knowledge. But they do need to understand how to manage well the people who do. Only the managers can provide the internal communication network (or at least support its evolutionary development) that will give the technical people the maximum benefit of each other's information and knowledge.

Organizations, in their most abstract form, are also networks. Some of the general mathematical and computational tools for analyzing networks have found recent application in the kinds of social and communication networks that impact human performance. The well known "small world" effect has been explained as phenomena that arises only in networks that are partly ordered and partly random, and quantitative bounds have been established that characterize the degree of "information contagion" in these networks as it affects group or team decision-making. (Watts and Strogatz 1998)

In any particular organization, considerable benefit may be derived from knowledge developed in one part of the organization if only it can be transferred effectively to help another part. Overcoming communication barriers within organizations can be very difficult. Sencorp uses a fractal model of information, knowledge and decision-making within its large conglomerate structure to overcome these barriers. (Personal Communication) One EPRI project used the methods of object-oriented analysis to produce an "Integrated Knowledge Framework" for a coal-fired power plant that showed explicitly where such cross over knowledge could be beneficial. (Wildberger 1997).

Emergency Response and the General Public
When all else fails and there has been a general collapse of a major part of the infrastructure, emergency response may be called for. Failure of the infrastructure network may only be concomitant to a natural disaster, like an earthquake or flood. But it is important that emergency response actions not further degrade the telecommunications, electric power, gas and water pipelines, etc., all of which can be of great assistance in mitigating the effects of the disaster. The usefulness of computer simulation has been demonstrated for both the analysis of emergency situations and for training people in their management. These simulations provide realistic representations in real-time of the damage caused by various natural and human-caused disasters. They can also provide statistical estimates of the results of human attempts to control this damage, for instance, the effects of a selected method for fighting a fire. But often, the greatest source of uncertainty in the management of emergencies is the behavior of the people directly affected by it. The training of emergency response teams emphasizes systematic, dependable performance even, if necessary, at the expense of rapidity and flexibility. Such dependability can come only after many repetitions of the training, to the point where details of behavior are almost automatic. This level of dependability is essential so that the plan for emergency management can safely assume that the team will perform as required. However, the behavior of the general population directly affected by the emergency is much less dependable. Emergency managers have

little basis for predicting it, and only limited ability to control it.

Past experience with other emergencies of a similar kind, sometimes even in the same geographic area, can provide rough guidelines. A time history including sequences of equipment failure and patterns of human response can now be assembled from the modern instrumentation being installed in most infrastructure networks as well as from direct reports by volunteers calling in on cellular phones. Using these collected space-time histories to forecast the behavior of the affected individuals may be possible using fuzzy hierarchical clustering. (Wildberger 1994)

Network Visualization and Situational Awareness

Exactly what they need to know may be different, but operators, managers, users and the general public all need to understand what is going on in the infrastructure network. Adequate visualization of the state of the system is required for situational awareness. The proliferation of new technology for multimedia user interfaces, and for virtual reality (VR) in particular, needs to be fitted into this context of human performance.

The information explosion has made attention an extremely valuable commodity for all workers and second only to capital in scarcity for managers. Interfaces should be designed so that the user will remain tuned in to many different factors while giving active attention only to a few. A variety of ways to display the behavior of the Internet have been suggested. Improved displays of the state of the electric power grid are been installed in control centers. (Anderson *et al.* 1994.) (Christie and Mahadev 1994) EPRI has been exploring the use of parallel coordinate transformation (Inselberg and Dimsdale 1991) to display very high dimensional data in a way that can be rapidly assimilated by power plant or power system operator. (http://www.caip.rutgers.edu/~peskin/epriRpt/) But there is room for a great deal of imaginative innovation in this area. For instance, little use has been made of esthetic considerations in the design of interfaces, yet it is clear that we are attracted to, and seek to use more frequently, that which is esthetically pleasing. These considerations may be especially significant if disaster mitigation information needs to be passed to the general public via cable or broadcast television.

VR may be especially useful in planning and in training for maintenance or rapid repair work, especially in hazardous situations. It could be an inexpensive way to try different configurations and different special purpose tools before the need arose. NASA is already using virtual reality in place of replica training simulators for team building/training with members in distributed locations. Improved haptics are the most obvious requirement both in VR and multimedia in general. But with the amount of research and development being done in these areas, completely virtual sensual concordance may not be far away. (Leigh *et al.* 1999)

SUMMARY AND CONCLUSIONS

In the electric power industry and other critical infrastructures, new ways are being sought to improve network efficiency and eliminate congestion problems without seriously diminishing reliability and security. Achieving these objectives in light of their vulnerability to cascading outages, initiated by material failure, natural calamities, intentional attack, or human error, as well as demands posed by economic, societal, and quality-of-life considerations and the ever-increasing interdependencies between interconnected infrastructures offers new and exciting scientific and technological challenges.

In many complex networks, the human participants themselves are both the most susceptible to failure and the most adaptable in the management of recovery. National and international infrastructures are characterized by the many points of interaction among their operators, managers, users, and the general public. No one is “outside” the infrastructure. Human performance issues include: handling warnings and other interruptions, shared planning between humans and automata, managing users' behavior for their mutual benefit, rapid knowledge transfer within the managing organization, and situational awareness for all.

DoD and EPRI are jointly funding a Complex Interactive Networks/Systems Initiative (CIN/SI) whose objective is to produce a significant, strategic advancement in the robustness, reliability, and efficiency of the interdependent energy, communications, financial, and transportation infrastructures. Some of the areas being investigated by the university consortia include:

- *Robust Control:* Extending the theory of robust control beyond the relatively narrow problem of feedback control.
- *Disturbance Propagation in Networks:* Prediction and detection of the onset of failures both at the local and at the global level including instability failure mechanisms and thresholds for identifying their onset.

- *Complex Systems:* Theoretical foundation of complex interactive systems.
- *Dynamic Interaction in Interdependent Layered Networks:* Hierarchical and multi-resolution modeling and identification in networks.
- *Modeling in General:* Efficient simulation techniques and generic modeling research, including developing a modeling framework and analytical tools for studying the dynamics and the failure modes in the interaction of economic markets with power and transportation systems via a delay and loss prone communication network.
- *Forecasting Network Behavior and Handling Uncertainty and Risk:* Characterization of uncertainty in large distributed networks, stochastic analysis of network performance, and handling rare events through large deviations theory.

Six consortia, consisting of 28 universities, are focusing on advancing basic knowledge and developing breakthrough concepts in modeling and simulation; measurement, sensing, and visualization; control systems; and operations and management.

INFORMATION AND REFERENCES

CIN/SI Web Sites

The formal CIN/SI research announcement, DAAG55-98-R-RA08, can be reviewed at: www.aro.army.mil/research/complex.htm.

A CIN/SI presentation package is available at: http://www.epri.com/targetST.asp?program=83.

An EPRI news release is available: www.epri.com/corporate/discover_epri/news/releases/dod.html.

Reports from Related EPRI projects

Annual technical reports, detailing work of each consortium, are available as EPRI reports:

TP-114660: Complex Interactive Networks/Systems Initiative: Overview and Progress Report for Joint EPRI/Dept. of Defense University Research Initiative, (Author: M. Amin)

TP-114661: Conceptual Design of a Strategic Power Infrastructure Defense (SPID) System, (Authors: PI and Co-PIs at the U. of Washington, Arizona State U., Iowa State U., Virginia Tech.)

TP-114662: Intelligent Management of the Power Grid: An Anticipatory, Multi-Agent, High-Performance Computing Approach, (Authors: PI and Co-PIs at Purdue U., U. of Tenn., and Fisk U.)

TP-114663: Modeling and Diagnosis Methods for Large-Scale Complex Networks, (Authors: PI and Co-PIs at Harvard U., Boston U., MIT, U. of Massachusetts-Amherst, Washington U.-St. Louis)

TP-114664: Minimizing Failures While Maintaining Efficiency of Complex Interactive Networks and Systems, (Authors: PI and Co-PIs at Cornell U., George Washington U., UC-Berkeley, U. of Illinois, Washington State U., U. of Wisconsin)

TP-114665: Context-Dependent Network Agents, (Authors: PI and Co-PIs at Carnegie Mellon U., RPI, Texas A&M U., U. of Illinois, U. of Minnesota)

TP-114666: From Power Laws to Power Grids: A Mathematical and Computational Foundation for Complex Interactive Networks, (Authors: PI and Co-PIs at CalTech, MIT, UCLA, UC-Santa Barbara, U. of Illinois)

Further Reading and References

Amin, M. (2000) National Infrastructures as Complex Interactive Networks, Chapter 14 in *Automation, Control, and Complexity: New Developments and Directions*, Samad & Weyrauch (Eds.), John Wiley and Sons

Amin, M. (2000) "Energy Infrastructure Interdependencies: Challenges for R&D," 6 pp., Proceedings of the 8th Int'l Energy forum and the Global Energy Exposition, Las Vegas, July 23-28

Amin, M. and Holmes, R. (2000) "Adaptive Infrastructures – the Complex Interactive Networks and Systems Initiative," 9 pp., Proceedings of the ISA 2000 conference, San Antonio, TX, June5-9

Amin, M. (1998a) Complexity and the deregulation of the electric power industry, in *Proceedings of the 3rd Embracing Complexity (EC3) Conference,* Cambridge, MA, 101–106.

Amin, M., Wildberger, M., and McCarthy, G. (1998b) Self-healing electric power grid as a complex adaptive system, in *Proceedings of the 4th Joint Conf. on Information Sciences (JCIS'98) and the 2nd International Workshop on Intelligent Control,* Research Triangle Park, N.C., Vol. I, 295–301.

Amin, M. (ed.) (1999) *Operations Research Methods in Intelligent Transportation Systems, International Transactions in Operational Research,* **6**(1).

Amin, M., Garcia-Ortiz, A., and Wootton, J. (eds.) 1995 *Network, Control, Communications and Computing Technologies in Intelligent Transportation Systems,* (Guest co-editors: Amin, Garcia-Ortiz and Wootton), *Mathematical and Computer Modeling*, Elsevier Science Ltd, Vol. 22, No. 4-7, 454 pp.

Anderson, M.D.; H.J. Pottinger; C. Schroder; and H. Qiu. 1994. "Advanced Graphics for Power System Operation." EPRI Technical Report TR-103633.

Astrom, K. and Wittenmark, B. 1995. *Adaptive Control*, Addison Wesley.

California Energy Commission. 1997. "A Survey of the Implications to California of the August 10, 1996 Western States Power Outage," Report P700-97-003, June 1997.

California Independent System Operator. 1998. "1998 Transmission Reliability Report," prepared by the California Independent System Operator in consultation with The AB 1890 Report Steering Committee.

Cebrowski, A.K. and Garstka, J.H. 1998. "Network Centric Warfare – Its Origin and Future." In *U.S. Naval Institute Proceedings* (Jan.), 28-35.

Cobb, Adam, "Thinking about the Unthinkable, Vulnerabilities to High-Tech Risks", 29 June 1998, Parliament of Australia Research Paper 18 1997-98

Cowan, G.A., D. Pines, and D. Meltzer. Eds. 1994. *Complexity: Metaphors, Models, and Reality*. Addison-Wesley: Reading, MA.

Critical Foundations: Protecting America's Infrastructures, The report of the President's Commission on Critical Infrastructure Protection, October 1997, Washington DC, www.ciao.ncr.gov

Christie, R.D. and P.M. Mahadev. 1994. "Visualizing Power System Data." EPRI Technical Report TR-102984.

Dick, AJ, "Security Implications of New Technology Metering and Associated Systems", 1998, UK Electricity Association

Dunn, R.S. and D. McBride. 1999. "Frequency Weighted Task Complexity Index, an Improved Metric for Operator Simulation and Man-Machine Integration." Department of Defense Human Factors Engineering Technical Advisory Group Meeting 42, May 10.

Dy Liacco, T.E. 1967. "The Adaptive Reliability Control System." IEEE on PAS, May, pp. 517-561

EPRI. 1994. Evolutionary Computing. Technical Brief TB-104097, Pleasant Hill, CA: EPRI Dist. Ctr.

EPRI. 1996. Integrated Knowledge Framework (IKF) for Coal-Fired Power Plants. Technical Report TR-106211-V1/2/3, Pleasant Hill, CA: EPRI Dist. Ctr.

EPRI. 1997. "Computer Simulation of Adaptive Agents for an Electric Power Auction." Technical Report TR-106975, EPRI Dist. Ctr , Pleasant Hill, CA.

EPRI. 1997. Computing and Telecommunications (C&T) Solutions Quarterly, Vol. 3, No. 1

EPRI. 1999. Electricity Technology Roadmap 1999 Summary and Synthesis, July 1999, Palo Alto CA, Report CI-112677-V1, www.epri.com

EPRI. 1998. Internet Energy Management: A Security Analysis, Palo Alto CA, Report TR-111015

Friedman, D. and J. Rust, eds. 1993. *The Double Auction Market*. Addison Wesley, Reading, MA.

Inselberg, A. and B. Dimsdale. 1991. "Parallel Coordinates: A Tool for Visualizing Multivariate Relations." In *Human-Machine Interactive Systems* A. Klinger, ed. Plenum Publishing.

Leake, D., ed. 1999. "Distributed Continual Planning." *AI Magazine* Vol. 20, No. 4. (Winter) The American Association for Artificial Intelligence (AAAI), Menlo Park, CA.

Leigh, J. et al. 1999. "Visualization in Teleimersive Environments." In *Computer* Vol. 32, No. 12 (December) IEEE Computer Society, Los Alamitos, CA: 66-73.

McFarlane, D.C. 1999. "Coordinating the Interruption of People in Human-Computer Interaction," *INTERACT '99: Seventh IFIP Conference on Human-Computer Interaction*. International Federation for Information Processing, Laxenburg, Austria.

McFarlane, D.C. 1997. "Interruption of People in Human-Computer Interaction: A General Unifying Definition of Human Interruption and Taxonomy." NRL Formal Report NRL/FR/5510-97-9870. Naval Research Laboratory, Washington, DC.

Miller, C.; H. Funk and M. Hannen. 1997. "Task-Based Interface Management: A Rotorcraft Pilot's Associate Example." In *Proceedings of the AHS Crew Systems Technical specialists Meeting*. Philadelphia, PA (September 23-25).

Pareto, V. 1927. *Manuel d'Economie Politique*. 2nd ed., M. Giraud, Paris.

Perrow, C. 1984. *Normal accidents: Living with High-Risk Technologies*. Basic Books.

Tenner, E. 1997. *Why Things Bite Back*. Vintage Books: NY.

Watts, D.J. and S.H. Strogatz. 1998. "Collective Dynamics of 'Small-World' Networks." *Nature*, 393:440-442 (June 4).

Wildberger, A.M. 1994. "Forecasting Time Series Using Hierarchical Clustering." In *Proceedings of the 1994 European Simulation Symposium, ESS 94* (Istanbul, Oct. 9-12) A.R. Kaylan;, A. Lehmann and T.I. Oren, eds., Ghent, Belgium: 268-272.

Wildberger, A.M. 1997. "Object-Oriented, Integrated Knowledge Framework for Electric Power Plants" In *Proceedings of Object-Oriented Simulation Conference (OOS'97)*: 3-10.

Wildberger M. and Amin, M. (2000) "Human Performance Issues in the Security of the National Infrastructure," 6 pp., Proceedings of the ASTC2000 (Advanced Simulation Technology Conference), Washington, DC, April 16-20

Willinger, W. and V. Paxson. 1998. "Where Mathematics Meets the Internet." *Notices of the AMS*, Vol. 45, No. 8 (Sept.).

Section 1

Distributed Control & Agent Based Systems

Tactical Intelligence Tools for Distributed Agile Control of Air Operations

Shashi Phoha
Applied Research Laboratory
The Pennsylvania State University
University Park, PA 16802
sxp26@psu.edu

Natarajan Gautam
Industrial Engineering
The Pennsylvania State University
University Park, PA 16802
ngautam@psu.edu

Alan Horn
Applied Research Laboratory
Pennsylvania State University
University Park, PA 16802
ajh14@psu.edu

Abstract

This paper presents an innovative architecture for engineering an agile distributed system of interacting agents, which is modeled as a set of interacting automata[11]. By slight modification of Goranson's definition of agility [5], we define agile control as the collective ability of agents to continually adapt to expected and unexpected changes. The intelligent control of the enterprise, then, consists of (i) continual observation of events and identification of relevant changes on the battlefield (ii) controllers implementing a dynamic C^2 strategy as a control specification for each agent in the enterprise, and (iii) tactical intelligence tools which process necessary transient and situational information for the agents to execute the control decisions in the battlefield. A testbed implementing these components and their interactions has been established as shown in Figure 1. Methods for engineering agile control specifications for a hierarchy of agents in the air operations enterprise are discussed in a companion paper in these proceedings [14]. This paper presents the overall architecture and its testbed implementation for agile control of air operations. In addition, a variety of tactical intelligence tools are discussed which provide situational and transient information for executing control decisions. Algorithms for dynamic assignment of targets to aircraft is an example of a tactical intelligence tool. These algorithms require current and situational knowledge of resources and positions of identified targets to continually reconfigure assignments. Both centralized and onboard tactical intelligence algorithms are discussed and compared for relative performance. The testbed is being used for more detailed experimental verification of C^2 strategies for air operations as described in another companion paper in these proceedings [4].

1. Intelligent Control of Dynamic Operations

The air operations enterprise can be represented as a dynamic hierarchy of intelligent agents who change their internal states in response to interactions with other agents or the environment. The dynamic time-evolution of complex interactions of agents in the enterprise is inherently different from the continuous variable dynamics of physical processes. Unlike the evolution of physical processes which are constrained by physical laws, agent interactions are triggered by discrete events and are characterized by a large number of interacting processes under predictable and unpredictable disturbances. Hence there are no inherent physical laws to constrain system configurations other than natural limitations of human/machine comprehension, resources and ergonomics. Therefore, an accurate plant model, based on physical laws, cannot easily be formulated. Concurrent dynamic processes, embedded at each node of the system, interact in highly non-linear, time-varying stochastic and possibly inconsistent ways. Hence model based conventional control methods are inadequate. Alternate methods of designing controllers whose structure and outputs are determined by empirical evidence through observed input/output behavior rather than by reference to a plant model are necessary.

Several techniques for such non-linear controller design have recently been proposed in recent literature on Intelligent Control [6, 7, 10, 2, 9, 12]. Meystel [8] has proposed a nested hierarchical control architecture for the design of Intelligent Controllers. Albus [2] has also developed the Real Time Control Architecture in which sensor and processing, value judgment, world modeling and behavior generation subsystems interact to adaptively generate appropriate response behaviors to sensor observations and knowledge of mission goals. Brooks' subsumption architecture [3] for intelligent control is based on achieving increasing pre-specified levels of competence in an intelligent system by examining outputs of lower levels.

An innovative behavior based architecture for intelligent agile control of agents in the air operations enterprise is presented here. The mathematical representation of agent interactions as a cellular space of interacting automata was formulated in [11]. The uniqueness of this approach is that it allows a heterogeneous group of agents to self organize and coordinate composite behaviors to support the execution of desired global behaviors of the system. Desired system behaviors are enabled through the hierarchical control architecture of the system and represented as control specifications. Methods for the control specifications for an agent hierarchy and for checking the controllability and hierarchical consistency of the resulting system are discussed in [14].

In section 2 we present an overall architecture for implementing agile enterprise control. Section 3 discusses centralized versus distributed tactical intelligence. In Section 4, we present four different algorithms for dynamic target scheduling and routing of aircraft in a limited SEAD Scenario [13] for corridor clearance. Section 5 discusses resource bounded optimization and performance comparisons for the four algorithms. More detailed experimental verification of distributed C^2 strategies on this testbed is discussed in [4].

2. Agile Control Architecture

The essential components of Agile Control are shown in Figure 1. These are:

(i) Identification and communication of relevant change in the enterprise to appropriate agents.

(ii) Hierarchical controller for specifying and implementing a control strategy, and

(iii) Tactical intelligence tools for providing intelligent decision support for executing the control strategy.

The description and interaction of these components will be described in detail in this paper.

An enterprise simulator, representing air operations in the battlefield, generates events which are observed and responded to by intelligent agents. The design of the hierarchical controller assumes the strategic knowledge an experienced commander may have in specifying a control strategy for his agents for a particular mission. For example, in the corridor clearing scenario described later in this paper, the control strategy will be based on mission planning information which may not be specific in terms of the exact location of targets. In general, sufficient planning information is assumed so that our overall control strategy can be implemented.

Control specifications define the execution level behavior-modification rules for each agent in response to observed changes in the enterprise. These specifications are *ill-posed*, however, for the execution level dynamics

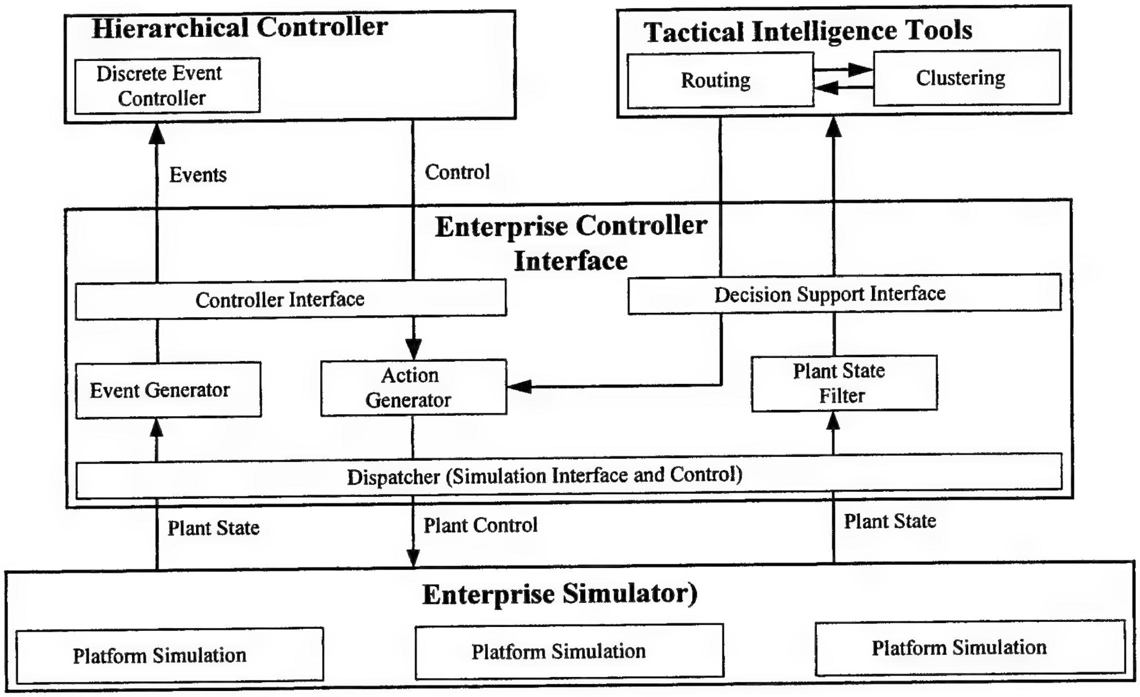


Fig 1. Components of the Agile Control Architecture

in the enterprise is not yet defined. Tactical intelligence tools are developed for providing situational and transient information derived from event observations (including status/position report) to support the execution of the control strategy. For example, upon identification of an unanticipated threat, the on-board controller may enable the 'escape' behavior for a fighter aircraft. The tactical intelligence tools on-board the aircraft must then provide move-to-coordinates (x, y, z) for safe escape. In general, the tactical intelligence tools identify observable change in the enterprise and provide situational decision support to the controller to enable an appropriate response. Together, the hierarchical controller and tactical intelligence tools define the response mechanism for an intelligent agent.

The agent interface to the enterprise allows the observation of change events in the enterprise by the agent and enables the control actions to be implemented in the enterprise.

This modular construction of the testbed allows multiple control algorithms to be tested for a given enterprise. It also allows variations in the plant simulator to evaluate control performance of a particular control strategy for various possible evolutions of the enterprise.

The following is a detailed description of the components in Figure 1:

Enterprise Simulator (ES) - The ES generates raw data for all forces, interactions among platforms, and environmental conditions. It also responds to inputs from the Dispatcher.

Dispatcher (D) – D receives data from the ES and sends it to the Event Generator (EG) and Plant State Filter (PSF). It acts as a bridge between the controllers and the enterprise simulator (ES).

Event Generator (EG) - The EG abstracts discrete events from the continuous data supplied by D. It also sends an event list to the hierarchical controller.

Action Generator (AG) - The AG takes an event vector from the controller and determines what actions should be taken. The AG, sometimes using tactical intelligent tools, determines what changes in the plant (continuous world) are required to enact each command (discrete events).

Plant State Filter (PSF) – The PSF reads pertinent data from the plant for use by the Tactical Intelligence Tools.

Controller Interface – This is a bridge between the EG software and the controller software.

Decision Support Interface – This is a bridge between the PSF software and the Tactical Intelligence Tools.

Hierarchical Controllers (HC) – The HC interpret the list of events from the EG and from this list they send an action vector to the action generator. An action vector is a control request to enable or disable specific controllable events.

The HC also abstract from event lists, sending these "higher-level" events to higher-level controllers. These higher-level controllers will also send commands back to the lower-level (original) controller through an action vector.

Tactical Intelligence (TI)– These are a set of tools used by the Action Generator to make optimized decisions. These contain algorithms for aircraft routing and creating target lists for aircraft, called clusters.

3. Centralized Vs. Distributed Tactical Intelligence

Some of the algorithms for tactical intelligence need to be performed in a centralized fashion, such as the planning algorithms before the start of a mission. However once the mission starts there is a choice of running the tactical intelligence algorithms either centrally or in a distributed fashion. The advantages of a centralized algorithm are potentially superior solutions due to global optimization. The disadvantages of the centralized algorithms are: the local information regarding enemy targets cannot be used, the communication overheads, security problems during communications, failure of communications, complexity of problems, large solution time for problems, potential inability to recover from multiple back-to-back failures, etc. Since the merits of the distributed algorithms are compelling for air combat command and control which are initially developed as centralized, are distributed over the platforms for autonomous, unsynchronized execution.

4. Algorithms for Tactical Intelligence

The tactical intelligence module uses several algorithms to assist the controller in making intelligent decisions. One of the algorithms to facilitate dynamic target scheduling and routing is explained here. It is a distributed algorithm that each friendly aircraft uses to autonomously decide its target schedule and routes. Some of the instances when this algorithm is used during the mission are: when an unknown enemy target is spotted, when the enemy attacks, during a mechanical failure, when fuel runs out, when weapons run out, when new aircrafts are added, etc.

4.1 Dynamic target scheduling and routing methods

Given the location of the friendly aircraft (e.g. Wild Weasel) and the expected locations of the enemy targets (with types such as fixed SAMS, mobile SAMS and radars) the objective of the dynamic target scheduling and routing algorithm is to compute a least-cost path to destroy all known enemy targets. The cost of a path is a function of the expected time to traverse a path and the expected risk incurred by the friendly aircraft. The following are the notations used in the algorithms:

N	:	Set of enemy targets
(x_i, y_i)	:	Coordinates of enemy target i ($i \in N$)
(x_0, y_0)	:	Location of friendly aircraft at the time of determining route
C_{ij}	:	Cost for directly traveling from enemy target i to enemy target j ($i, j \in N$)
R	:	An ordered set corresponding to the route of friendly aircraft
Z	:	Total cost for traversing route R
D	:	Dummy subset of enemy targets

Note that C_{ij} is computed using $(x_i,y_i),(x_j,y_j)$, speed of the aircraft, and, a risk factor (r_{ij}) such that among all paths between i and j, C_{ij} is cost associated with the path corresponding to the minimum of the product of the risk factor and travel time. We assume in this paper that C_{ij} has been pre-computed and is available to use in the algorithms. In a follow up paper we will describe algorithms to compute costs C_{ij} as well as the paths between targets i and j. Some algorithms for dynamic target scheduling and routing are now explained.

Greedy Algorithm: Given a set of enemy targets to destroy and the current location of the Wild Weasel, a greedy algorithm called the nearest neighborhood search algorithm is developed to obtain a sequence and route to destroy the targets. The greedy algorithm begins with the current location of the Wild Weasel and destroys the enemy target that can be reached by traversing the least cost path. Among the undestroyed targets, the algorithm next selects the enemy target that can be reached by traversing the least cost path. This process continues until all known enemy targets are destroyed. Then the Wild Weasel patrols its assigned region. The main advantage of using the greedy algorithm is to obtain high responsiveness for the tactical intelligence module. Hence the sequence and route to destroy the targets can be obtained extremely fast. However the price to pay for this high-speed response is the quality of the response. In most cases this response from the tactical intelligence is far from optimal, hence the overall objective of minimizing the cost of destroying the targets will not be accomplished.

<u>Algorithm Greedy</u>

```
1.  Set $i = 0$, $Z = 0$, $R := \varnothing$ and $D := N$
2.  While $D \neq \varnothing$, do
      a. $Z = Z + \min_{j \in D} C_{ij}$ and $k = \arg\min_{j \in D} C_{ij}$
      b. $R = R \cup k$ and $D = D - k$
      c. $i = k$
```

Resource Bounded Optimization (RBO) Algorithm: Given a set of enemy targets to destroy and the current location of the Wild Weasel, an RBO algorithm based on the 2-opt search for the well-known graph theoretic problem, the traveling salesperson problem, is developed. Note that the problem of obtaining a sequence and route to destroy the targets can be stated as a Hamiltonian path problem as explained in the algorithm that follows. In the RBO algorithm, the solution from the greedy algorithm is taken and improvised. At each iteration, a random-pair-wise interchange to the current sequence and route to destroy the targets is performed. After several iterations, the algorithm will converge to the optimal solution. The number of iterations the tactical intelligence will perform will depend on the time available to respond. The solution quality will improve with the number of iterations. This algorithm can be stopped at any iteration and a solution can be obtained. The numerical examples indicate a vast improvement in the solution quality (as compared to the greedy algorithm and the optimal algorithm) in very few iterations. However the disadvantage is that it may take a long time to obtain the optimal solution especially in ill-posed problems. We require a few definitions for the algorithm. Consider an element i in the ordered set R. Define $P(i)$ and $S(i)$ as the elements preceding and succeeding i respectively in the ordered set R. In other words $P(i)$ and $S(i)$ are respectively the enemy targets destroyed before and after destroying target i. Note that if i is the first or last target respectively, then $P(i)$ or $S(i)$ are null sets. Also define an operation $U(A)$ on a set A that results in a 2-tuple denoting 2 elements randomly selected from set A. Denote the binary variable $Y(i,j)$ such that it is 1 if arc (i,j) is in route R and, 0 otherwise.

<u>Algorithm RBO(I)</u>

```
1. Set $I$ as the number of desired iterations
2. Obtain $R$ using Algorithm Greedy
3. Update $R$ to include the initial location of friendly aircraft, i.e., $R = \{0\} \cup R$
4. For $x = 1$ to $I$, do
     a. $(i, j) = U(R)$ [Note: $i$ and $j$ are 2 nodes randomly selected from the route]
     b. if $i \neq j$, $P(i) \neq \varnothing$, $P(j) \neq \varnothing$, $S(i) \neq \varnothing$, $S(j) \neq \varnothing$, $P(i) \neq j$, and $P(j) \neq i$
     i.   $n1 = i$, $m1 = S(i)$
     ii.  $n2 = j$, $m2 = S(j)$
     iii. if $C_{n1,m1} + C_{n2,m2} > C_{n1,n2} + C_{m1,m2}$,
            then modify $R$ such that
              $Y(n1,m1)=0, Y(n2,m2)=0, Y(n1,n2)=1$ and
              $Y(m1,m2)=1$
            elseif $i \neq j$, and $S(i) \neq \varnothing$,
              $n1 = i$, $m1 = 0$ and $n2 = j$, $m2 = S(j)$
            if $C_{n2,m2} >. C_{n1,2}$,
            then modify $R$ such that
```

```
        $Y(n2,m2)=0$ and $Y(n1,n2)=1$
      elseif $i \neq j$, and $S(j) \neq \emptyset$,
        $n1 = i, m1 = S(i)$ and $n2 = j, m2 = 0$
      if $C_{n1,m1} > C_{n1,n2}$,
      then modify $R$ such that
        $Y(n1,m1) = 0$ and $Y(n1,n2)=1$
   else
go to a.
```

Note that other "elseif" conditions can be incorporated into the algorithm. For presentation reasons those conditions have been omitted.

Hamiltonian Path: The path traversed by the Wild Weasel from the current location to the last destroyed target is referred to as the Hamiltonian path in graph theory literature (see Ahuja [1]). The algorithm uses a network representation such that the known enemy targets are the nodes of the network. There is an arc from every node to every other node denoting the ability to go from any target to any other target. The cost on the arc represents the cost of traversing from one target to another and is computed using the expected travel time and expected risk. The algorithm begins by solving the minimal spanning tree (another well-known graph theoretic problem) of the network. If the spanning tree is not a Hamiltonian path, then it has arcs that violate the Hamiltonian path requirements. The solution of the minimal spanning tree acts as the lower bound. Then at every iteration, the lower bound is improved using a branch-and-bound technique where one of the violating arcs of the spanning tree is set to a high cost. The algorithm stops when a Hamiltonian path is obtained, i.e., there are no more branches to consider. This algorithm can guarantee optimal solution after a sufficiently large number of iterations. The major drawback is that if the algorithm is stopped during any iteration, no solution will exist that can be responded by the tactical intelligence.

We introduce a few notations before illustrating the algorithm. Let $C = [C_{ij}]$ be a cost matrix denoting the arc cost between every pair of nodes. Define $T(C)$ as an operation on C that results in a minimum spanning tree of the network. In particular, let $X = T(C)$ such that $X = [X_{ij}]$ and X_{ij} is a binary variable such that it is 1 if arc (i,j) is in the spanning tree and, 0 otherwise. For minimum spanning tree algorithms, see [1] to determine if a spanning tree is a Hamiltonian path is to check if the degree of all nodes in the spanning tree is not greater than 2, except node 0 whose degree should not be greater than 1. In essence, the following algorithm iterates through several spanning trees until a Hamiltonian path is obtained.

Algorithm Hamiltonian

```
1. Set $W = 0$ and $X = T(C)$
2. If $X$ is a Hamiltonian path, $W = 1$
3. While $W = 0$, do
   a. $Z = 0.5\,XC$, the total cost of the spanning tree
   b. For every node with degree greater than allowed:
      i.  Obtain $X = T(C)$ assuming one of arc cost of
          the node is infinite
      ii. If $X = T(C)$ is not a Hamiltonian path, go to
          b. Else $Z_0 = 0.5XC$
   c. Choose Hamiltonian path with smallest $Z_0$
4. $Y = X$ and obtain the route $R$ via $Y$.
```

Note that it is possible to improve the above branching procedure by doing a branch-and-bound procedure. This would require fathoming all minimum spanning trees whose costs are greater that the current Hamiltonian path cost during an iteration. If the algorithm is stopped in the middle, to avoid the situation of not having a route we do the following: if a (current) Hamiltonian path is available it can be used, otherwise, the solution from the greedy algorithm can be used.

Optimal Algorithm: This algorithm uses complete enumeration of the entire solution space to obtain the optimal solution. Therefore every possible sequence and routes to destroy the targets are considered and the best is chosen. This is a very time-consuming algorithm and takes $n!$ computations if there are n targets to be destroyed. Since this algorithm guarantees an optimal solution, it is used for benchmarking other algorithms.

Define $\overline{R}$ as a set of all possible routes and r as a candidate route. Also let $D(r)$ be the cost of the candidate route r.

Algorithm Optimal

```
1. Set $Z = \infty$
2. For all $r \in \overline{R}$, if $D(r) < Z$, then $R = r$ and $Z = D(r)$
```

5. Resource Bounded Optimization Performance Comparisons

The algorithms developed in the previous section are tested here for different numerical values. In particular, the performance of the new RBO algorithm developed is tested against the other algorithms for an air operation scenario described below.

5.1 Generic Air Ops Scenario

The scenario considered here is a limited SEAD scenario. A bombing mission is to be attempted against an enemy airbase. For the bombers to be able to perform the mission, enemy air defenses must be disabled in two corridors leading to the base. In the scenario, the corridors are given. They are four miles wide at their narrowest, to insure the safety of aircraft flying down the middle of the corridor. The enemy has three types of entities: (1) fixed SAM sites, (2) mobile SAM launchers, and (3) fixed radar sites. The mobile SAM launchers perform a random walk on the terrain. They stop wandering and prepare to attack at random points in the walk. Any target that has been hit is disabled for a random period of time.

Friendly forces are limited to Wild Weasels, which search for and destroy SAMs. Each Wild Weasel has its own discrete event controller. The local controller has access only to local information. Another discrete event controller coordinates activities among the Wild Weasels. The coordinator interacts with the system by receiving ISR inputs and sending radio messages to the aircraft. Each aircraft starts with an initial mission to be completed. The aircraft's controller determines the decisions that are taken as events occur. Missions will be to patrol parts of the corridor and destroy enemy entities. Aircraft communicate with the supervisory controller as needed for coordination. It will adjust the regions covered and targets aircraft in response to changing conditions.

The tactical intelligence module explained in Section 2 is responsible for (1) allocating platforms to targets and regions (2) allocating routes to platforms, and, (3) allocating patrolling pattern after destroying known SAMs in a region. In this paper we have explained in detail the algorithms only for the second task, i.e. allocating routes to platforms. The other algorithms are explained in [4]. It is important to note that these algorithms are executed both during the initial planning phase as well as en-route during the attack phase. Therefore it is critical to obtain an algorithm that produces reasonably good results in a short period of time. At this time, we only compare the performance of the algorithms running independently. However, in a future paper, we will provide the results based on battlefield simulations.

The coordinator assigns regions for aircraft to cover. This is a centralized algorithm that uses a clustering algorithm (K-Means algorithm) and regions are created by a Vornoi process which partitions a corridor into disjoint regions. The individual aircraft choose their own strategies for destroying known targets based on a decentralized algorithm. Once all known targets are destroyed, another decentralized algorithm (such as a lawnmower-type algorithm) to patrol for new threats is used. Regions must be reassigned and strategy for destroying targets must be reformulated as aircraft are destroyed, unknown enemy targets are spotted, aircraft run out of fuel or weapons, or new aircrafts are added.

5.2 Performance Metrics

The dynamic target scheduling and routing methods are considered here. It is assumed that regions have been described and targets have been assigned to each aircraft. In order to compare the four algorithms in Section 4.1, we use two performance metrics: solution quality and number of floating-point operations. The *solution quality* is benchmarked against the best possible solution. Therefore the ratio between the optimal solution (produced using the optimal algorithm) and the solution produced by an algorithm is the measure considered for solution quality. The *number of floating-point operations* is a measure of the number of operations that will be required on a computer to obtain the given solution. Then based on the type of computer installed on the aircraft, this metric can be used to determine the time to respond to the controller with a route.

5.3 Performance Evaluation

To evaluate the performance of the different algorithms, using 30 sets of enemy target locations to destroy and current location of the Wild Weasel, for each set, the following algorithms were considered and average performance metrics were obtained: greedy algorithm, RBO(I) algorithm with I=5, 10, 25, 50 and 100 iterations, Hamiltonian path algorithm (which is stopped after a sufficient number of iterations), and, optimal algorithm. The performance metrics are tabulated in Table 1, below. This table is based on 8 enemy targets assigned to an aircraft. Any value larger than 8 targets would require very large computational time for the optimal solution. However, for the other algorithms we could use many more targets. Also, the table is obtained by running the algorithms off-line. From the table note that the RBO algorithm with just 10 or 25 iterations results in a good solution. Therefore, if the RBO algorithm needs to be aborted after, say 5000 floating point operations, the solution obtained is very good. On the other hand, the Hamiltonian algorithm after 5000 floating point operations would not have produced any solution. Also, the greedy algorithm would have used an inferior solution. The RBO algorithm produces significantly better results than the greedy algorithm in a very few extra iterations. Therefore for the SEAD scenario it would be most appropriate to use the RBO algorithm and depending on the time available to solve the RBO, the algorithm can be stopped at a suitable time.

	Greedy	RBO (5 itns)	RBO (10 itns)	RBO (25 itns)	RBO (50 itns)	RBO (100 itns)	Hamilt.	Optimal
Solution quality	0. 9575	0.9683	0. 9727	0.9824	0. 9845	0. 9868	0. 9907	1.0000
Floating pt. Ops.	992	2589	4095	8601	15898	30395	49048	351769

Table 1. Performance metrics

Acknowledgements and disclaimers

This effort is sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-99-1-0547 (JFACC). The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

References

[1] R.K. Ahuja, T.L. Magnanti, and, J.B. Orlin, *Network Flows: Theory, Algorithms and Applications*, Prentice-Hall Inc., 1993.

[2] J.S. Albus, "A Reference Model Architecture for Intelligent Systems Design," in *An Introduction to Intelligent and Autonomous Control*, pp 27-56. Kluwer Academic Publishers, 1993.

[3] R.A. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE Transactions on Robotics and Automation*, 2(3): pp 14-23, 1986.

[4] R. R. Brooks, C. Griffin, P. Dicke, M. Byrne, M. Edwards, S. Phoha, D. Friedlander, B. Button and E. Grele, "Experimental Verification of Distributed C^2 Strategies," proceedings of 2nd DARPA JFACC *Symposium on Advances in Enterprise Control*, Minneapolis, MN, July 10-11, 2000.

[5] H. T. Goranson, *The Agile Virtual Enterprise Cases, Metrics, Tools*, Quorum Books, 1999.

[6] C. J. Harris, ed., *Advances in Intelligent Control*, Taylor & Francis, Bristol, PA, 1994.

[7] A. H. Levis, "Modeling and Design of Distributed Intelligence Systems, " in *An Introduction to Intelligent and Autonomous Control*, pp 109-128, Kluwer Academic Publishers, Boston, M, 1993.

[8] A. Meystel, "Autonomous Mobile Robots: Vehicles with Cognitive Control, "*Proceedings of the World Scientific*, Singapore, 1991.

[9] S. Phoha, S. Sircar, A. Ray, and I. Mayk," Discrete Event Control of Warfare Dynamics," The Technical Proceedings of the *1992 Symposium on Command and Control Research and the 9th Annual Decision Aids Conference*, Monterey, CA, June 8-12, 1992.

[10] S. Phoha, E. Peluso, P.A. Stadter, J. Stover, and R. Gibson, "A Mobile Distributed Network of Autonomous Undersea Vehicles," Proceedings of the *24th Annual Symposium and Exhibition of the Association for Unmanned Vehicle Systems International*, Baltimore, MD, June 3-6, 1997.

[11] S. Phoha and R. Brooks, "A Constructivist Theory of Distributed Intelligent Control of Complex Dynamic Systems," DARPA JFACC *Symposium on Advances in Enterprise Control*, San Diego, CA, November 15-16, 1999.

[12] P. J. Ramadge, W. M. Wonham, "Supervisory Control of a Class of Discrete Event Processes," *SIAM J. Control and Optimization*, Vol. 25, No. 1, January 1987.

[13] SEAD Scenario. http://www.cgi.com/web2/govt/seadystorm, April 7, 2000.

[14] W. Xi, A. Ray, S. Phoha and W. Zhang "Hierarchical Consistency of Supervisory Command and Control of Aircraft Operations," proceedings of the 2nd DARPA JFACC *Symposium on Advances in Enterprise Control*, Minneapolis, MN, July 10-11, 2000.

Experimental Verification of Distributed C^2 Strategies

Michael Byrne
mab374@psu.edu

Christopher Griffin
csg286@psu.edu

Shashi Phoha
sxp26@psu.edu

Eric Grele
grele@psu.edu

Richard R. Brooks
Applied Research Laboratory
Penn State University
P.O. Box 30
State College, PA 16804
rrb5@psu.edu

Marcus Edwards
mde11@psu.edu

Philip Dicke
pjd130@psu.edu

David Friedlander
dsf10@psu.edu

Brian Button
bab200@psu.edu

Abstract

This paper describes experimental approaches to quantifying advantages of distributed command. Van Creveld's [10] study of command and control concluded that distributed command and control systems are more effective. We hypothesize that C^2 hierarchies should be implemented as an interacting network of Discrete Event Dynamic System (DEDS) controllers. The network adapts dynamically to the operation. In this paper, we describe experimental approaches for verifying this thesis. They include ongoing work on a corridor clearing suppression of enemy air defense experiment. Control specifications modeling the C^2 strategies are described in a companion paper [14]. Tactical intelligence and coordination can be preformed centrally, by a fixed distributed hierarchy, or by a fluid distributed hierarchy. Detailed implementation of the testbed and essential tools for tactical intelligence are discussed in [8]. This paper documents experiments that explore aspects of the adaptive DEDS hierarchy. These aspects include sensitivity to partial observability, and the importance of coordination among platforms.

Keywords: JFACC, Distributed Control

1. Problem description

Combat is competition between opposing forces. Each force has its own objective, which may not be known by the opponent. Success requires adaptation to changing circumstances. Adaptation requires accurate knowledge. Early writings [9] emphasize that success in war depends on accurate knowledge of enemy and friendly forces including their abilities and current state.

Successful tactical warfare relies on adaptation to conditions on the battlefield [11]. This process can be put into the form of a feedback control loop. Formulation and implementation of control specifications and tactical intelligence for dynamic control of plant evolution are discussed in other papers in these proceedings [14, 8]. Friendly forces (controller *K*) receive information (ex. reconnaissance) from the environment (plant *P*), change their plans, and perform actions on the battlefield. *K* may consist of multiple controllers.

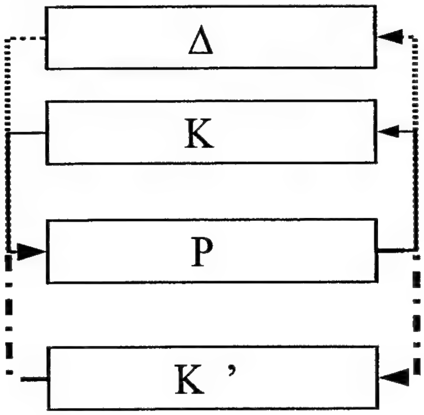


Figure 1. Opposing controller feedback model with noise

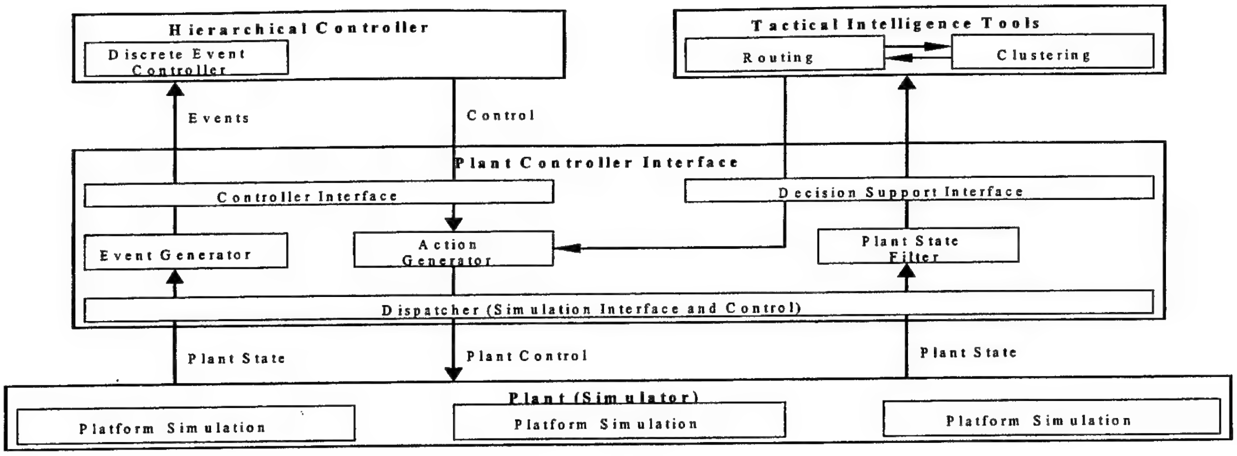


Figure 2. Block diagram of simulation framework used.

An alternative presentation, removes enemy forces from the plant. They become an opposing controller (*K'*), which attempts to force the plant into a state not desired by the friendly controller (*K*). Robust control theory includes system noise (*Δ*) [15] as in figure 1. One of the most influential studies of military strategy is from von Clausewitz [10]. He defines two inescapable influences on war: *fog* and *friction*. *Fog* is the inability to know the current situation on the battlefield. This is equivalent to sensor noise. *Friction* refers to the fact that the results of any action taken will not be the results intended. This is equivalent to transducer noise in a feedback loop.

History inspired Van Creveld to define five characteristics hierarchical systems need to adapt to the chaotic condition of battle [11,4]:

- Decision thresholds fixed far down in the hierarchy.
- Self-contained units exist at a low level.
- Information circulates from the bottom up and the top down.
- Commanders actively seek data to supplement routine reports.
- Informal communications are necessary.

Military command and control (C^2) is hierarchical and distributed, relying on feedback at many levels and time scales. No single fixed hierarchy can be appropriate for every mission, since "fog" and "friction" [10] make it impossible to accurately predict the course a battle will take in advance.

This paper is organized as follows: Section 2 summarizes the detailed descriptions in [8] for the testbed developed for studying decentralized problems composed of co-evolving agents. Section 3 compares our model with current economics studies contrasting centralized and decentralized organizations. In sections 4-6, we present results from simple simulations involving two aircraft and an air defense target. Section 7 presents a formal description of our central hypothesis and the scenario simulated to test the hypothesis. Section 8 provides results from large-scale simulators. Section 9 describes our conclusions and outlines topics for future research.

2. Simulation based evaluation of co-evolution

We model the C^2 hierarchy in terms of interactions among multiple decision-makers as described in [7]. Each C^2 node is an Independent Discrete Event Dynamic System (agent). Actions taken by an agent depend on its internal state, inputs from the environment, and inputs from other agents.

An open question is how to best derive and evaluate behaviors in systems consisting of multiple interacting agents. Interactions among individual agents resemble an ecosystem of evolving entities [6]. This is especially true in military systems where adversaries often provide deceptive information and decisions must be made in a timely manner. Agent based simulations rely less on unrealistic assumptions than analytic methods. While they may not converge to a unique strategy, they are likely to converge to more robust results [6].

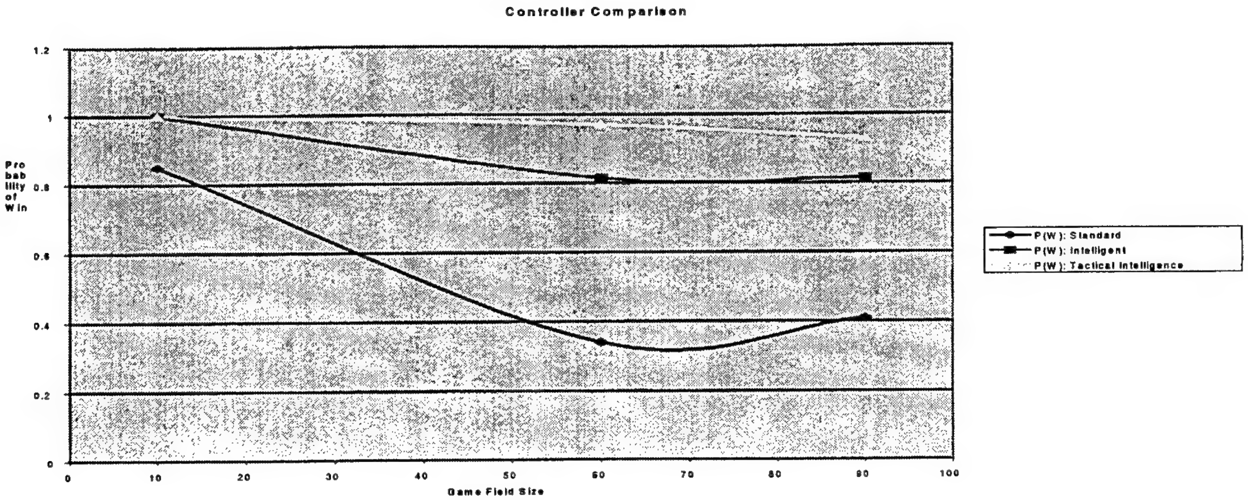


Figure 3. Results from two aircraft one-target scenarios.

A methodology for evaluating C^2 hierarchies based on simulations using interactive agents has been developed in [7]. Agents are personified by Discrete Event Controllers. Figure 2 is a block model developed in detail in [8] of testbed implementing this architecture. The plant consists of an air operations simulation. The hierarchical controller is a cellular space of interacting DEDS controllers. Tactical intelligence tools augment the DEDS controller. Each controller can access tools as needed. The plant controller interface translates from the continuous model implemented by the plant to the discrete model used by the controller. The event generator detects events of interest to the controller. The action generator translates controllable events output by the controller into commands the air operations simulation can execute. This approach allows all controllers to execute independently, exchanging information, and interact with the environment as they would in reality. Executing multiple runs using a Monte Carlo approach provides answers as to how the systems would adapt to their environment, and how interactions allow the system to co-evolve.

3. Economics model

The question as to whether centralized or decentralized control is superior is of general interest. While our work concentrates on military C^2, economists and social scientists have also studied this problem. Problems include insuring the consistency of results across multiple levels of fidelity and aggregation. A detailed study of this problem has been performed in [5].

The military has recognized the necessity of adapting civilian advances as appropriate, such as the use of network-centric coordination by Wal-Mart [2]. The central thesis in [3] is that a fitness landscape, a multi-dimensional space of consumer preferences, defines regions where decentralized control is superior to centralized control. Organizational adaptation and learning is driven by either a centralized (Ames) or decentralized (Wal-Mart) organization. The dimensions considered are: store practices, consumer preferences, market homogeneity, and market stability. Results of their study are:

- De-centralized control is better when local markets are different.
- De-centralization tends to perform better over long time horizons.
- Centralization performs better when customers are very sensitive to store practices.
- Centralization outperforms decentralization when market fluctuations are very large.
- Decentralized control is better when market fluctuations are strongly correlated.

Military C^2 differs from economics models in a number of ways, including: (1) The landscape tends to be less smooth with more non-linear interactions. (2) Relevant time scales tend to be much shorter. (3) The economic model used did not include adversarial relationships. Nonetheless, many of these lessons are worth considering. De-centralized control has an

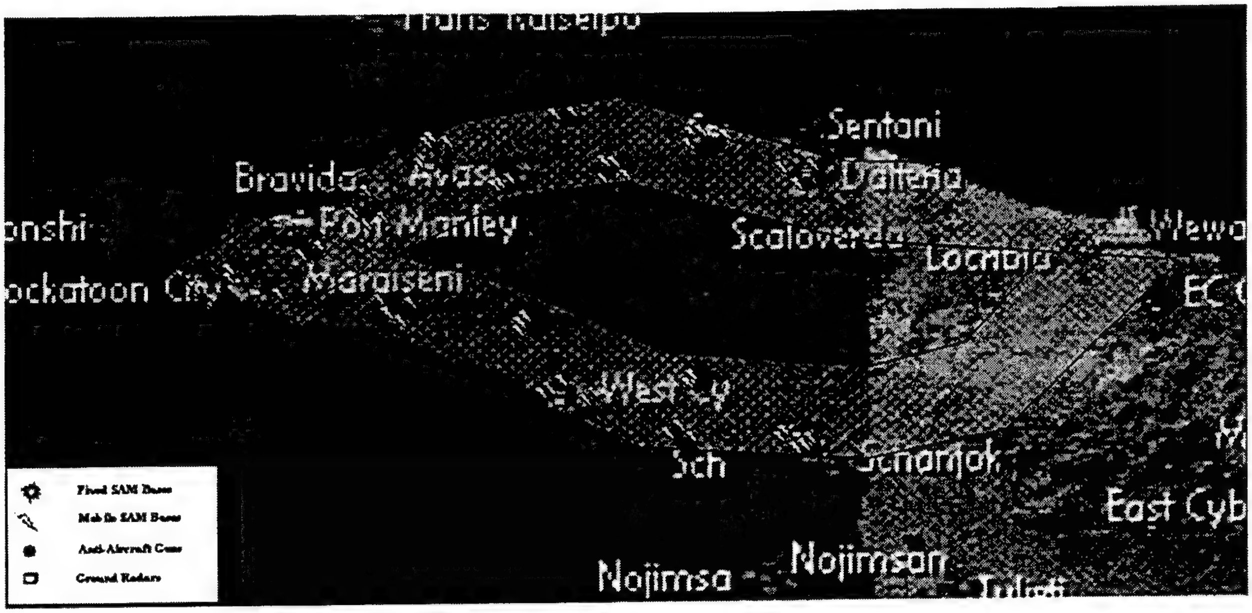


Figure 4. Two corridors need to be cleared between Wewok and Rockatoon city.

innate advantage by allowing better local adaptation to the fitness landscape. Centralized control can provide a short-term advantage by propagating information among units before the units have adapted to local conditions. When conditions fluctuate strongly, centralized control can find a more optimal global strategy, which is less sensitive to fluctuations than the local optima found by decentralized control.

4. Preliminary model

A prototype test bed incorporates a simple environment that tracks the status of multiple planes and targets in featureless terrains of varying size. It includes hierarchical information passing and tactical intelligence. We use probabilities of kill from [1] and restrict the altitude of the plane to 10,000 feet. We ran each experiment type on varying game field sizes. As the game board gets larger, the probability of successfully destroying the targets decreases. Additionally, the probability of successfully destroying all targets decreases as the number of targets increases.

Figure 3 presents results from a tactical intelligence controller, where two planes initiate a coordinated attack against a single target. The x axis is the size of the playing field. The y axis shows the probability of the wild weasel destroying the target. The bottom line is a graph of the proportion of wins by a set of planes using a normal controller with coordinated attack but not sharing information. The middle line shows the results of incorporating coordination into the plane controller. One plane informs the other when the target is detected. The top line allows the planes to make coordinated maneuvers in attacking the target. A higher level of intelligence increases the chances of success.

5. Controller partial observability

Figures 5, 6, and 7 illustrate results from experiments run simulating missions undertaken by a single Wild Weasel. In each mission a single Wild Weasel was sent to clear a region containing twenty five SAMs. The test ran until either the Wild Weasel was destroyed, all targets were destroyed, or five hours simulation time elapsed. The simulations ran to test the sensitivity of the individual DEDS controllers to incomplete information. Each scenario was run twenty times; discrete events were dropped at random during each run. The simulations show that the lower level controllers are robust to the loss of information.

Figure 5 shows the average number of targets destroyed versus the percent of events dropped. When perfect information was available, all targets were destroyed on all missions. When five percent or twenty percent of the events were dropped, the average number of targets destroyed remained high

(> 80%). When fifty percent of the events were dropped, only about 25% of the targets were destroyed. This performance is still acceptable, since it means that decisions were being made based on only half of the available information.

Figure 6 presents the probability that the wild weasel would be destroyed versus the percentage of events dropped. Once again, when no events were dropped the aircraft performed the mission perfectly. No aircraft were destroyed. The probability of the plane being destroyed appears to be more sensitive to information loss than the number of aircraft destroyed. When 5% of the events were dropped an aircraft had a 20% chance of being destroyed. When 20% of events were dropped, the probability of being destroyed reached 35%. Losing 50% of events caused the probability that the plane would be destroyed to reach 70%.

Time required to complete the mission appears less sensitive to information loss. With perfect information the corridor could be cleared within about two hours. The amount of time required climbed steeply as the amount of information available dropped. It then leveled off at slightly over three hours. This is probably due to the increased amount of time required to find targets, being offset by the increased mortality of the Wild Weasels.

These experiments show the DEDS controller to be robust to loss of information. Naturally, its performance degrades as the quality and amount of available information degrades. The amount of time required for the mission was not very sensitive to the amount of information available. The number of targets destroyed appears less sensitive to event dropping than the Wild Weasel's ability to survive.

6. Partial observability of specific events

Another set of partial observability experiments were run where specific types of events were dropped. Table 1 specifies the events understood by the controller and their meaning. Simulations like those described in section 5 were run, with the exception that only events of a specific type were masked from the controller.

Figure 8 shows the percentage of friendly wild weasels killed versus the percentage of specific event types masked. Clearly event type "A," the alarm event, is the most important event for determining whether or not a plane can survive. It signals when a plane is within the striking range of an enemy target. Event types "d" (plane destroyed) and "e" (escape from danger) also appear to influence a plane's survival, but only when 20% of the events are masked and not 50%. These two readings do not appear reasonable. We currently attribute them to experimental error, and are further investigating why this occurred.

Figure 9 shows the number of targets killed by a wild weasel on the average versus the percentage of

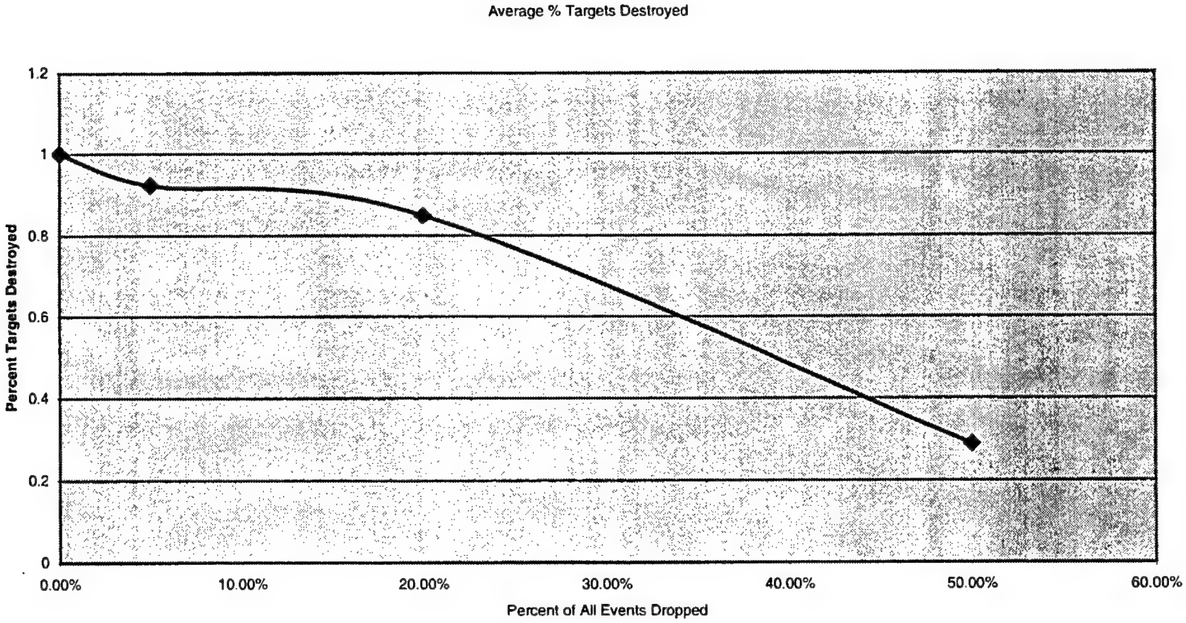


Figure 5. Percent targets destroyed versus percent of events dropped.

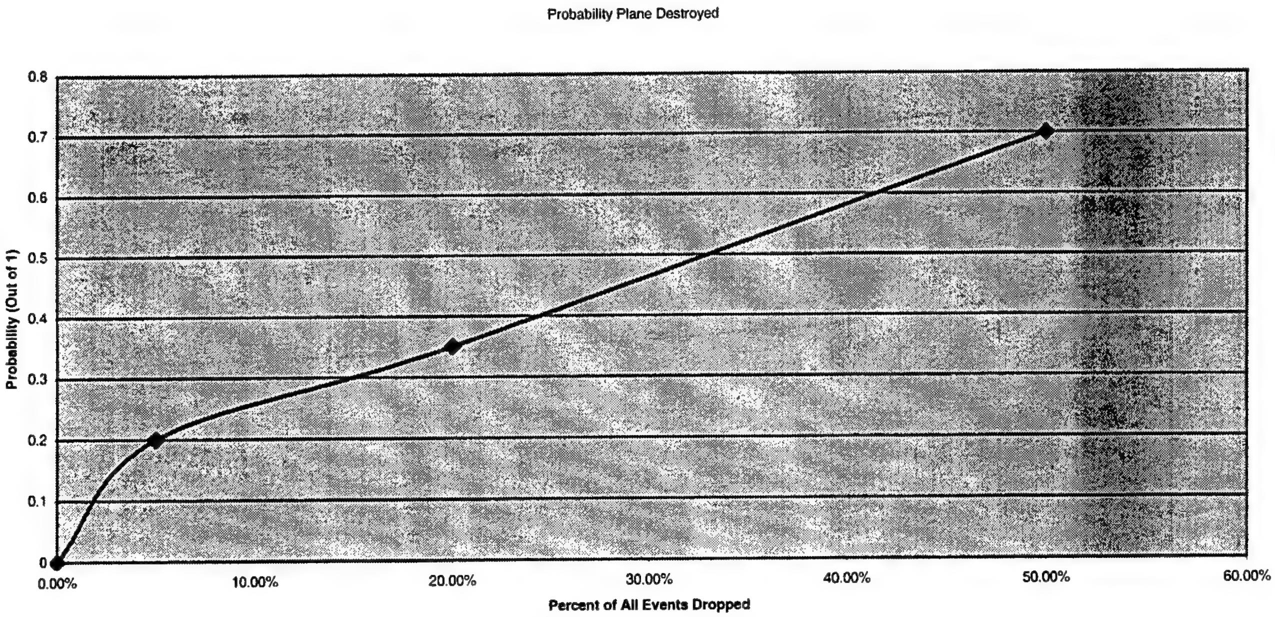


Figure 6. Probability that a Wild Weasel will be destroyed versus percent of events dropped.

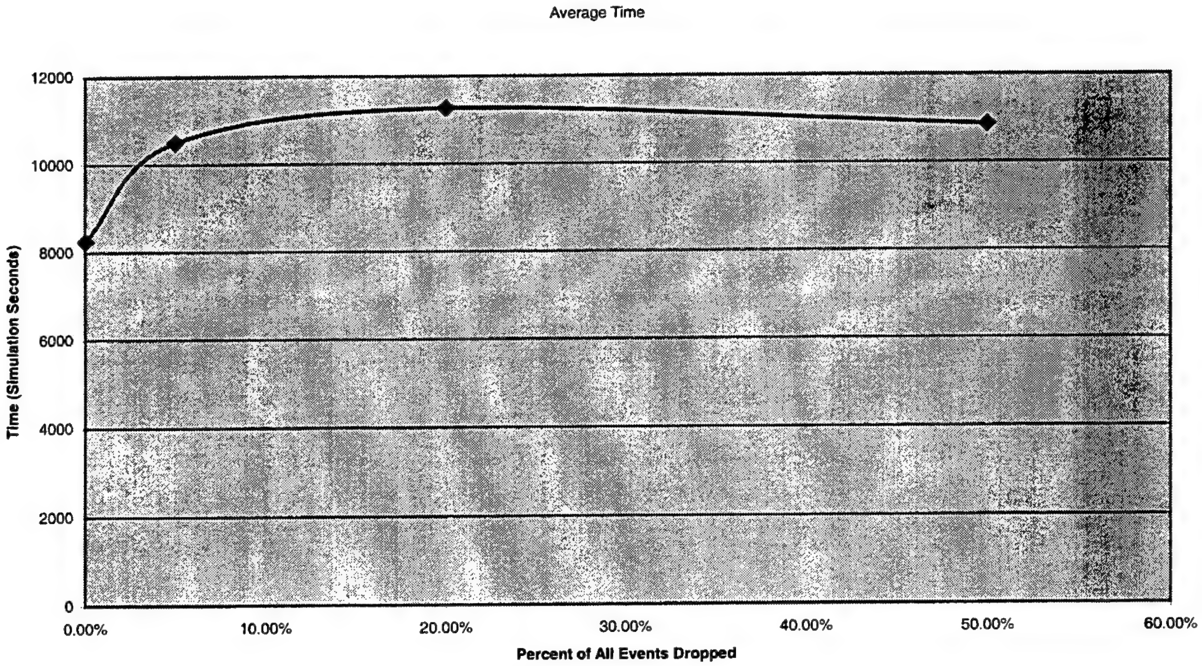


Figure 7. Time required to complete the mission versus percent of events dropped.

Table 1. Events in the discrete vent controller and their interpretation

Events	Physical Meaning
a	Fired a missile at the target
A	Plane enters enemy's firing range
b	Plane damaged (can fly, not fight)
C	Mission complete
d	Plane destroyed
D	Target destroyed
e	Plane moved to safe location
l	Plane's target list is empty
S/s	Plane moves to closest target
t	Take off

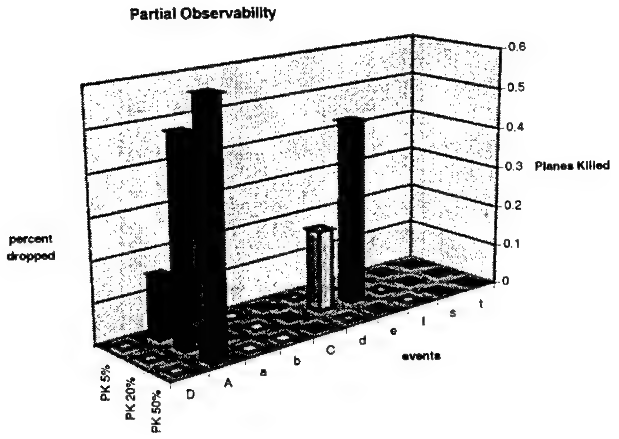


Figure 8. The percentage of wild weasels destroyed versus the number of percent of a given event type masked.

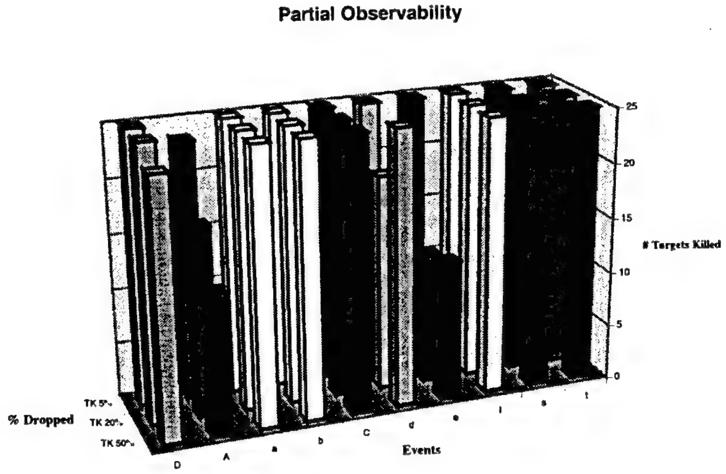


Figure 9. The number of targets destroyed versus the number of percent of a given event type masked.

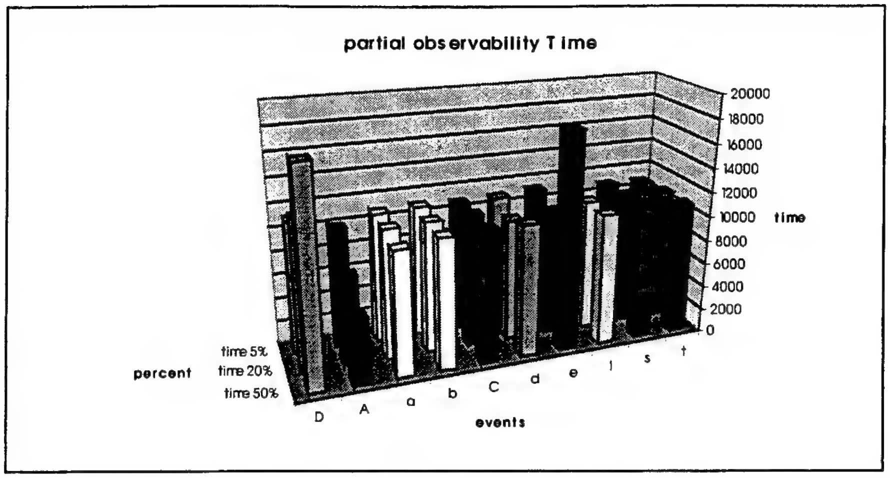


Figure 10. The amount of time required versus the number of percent of a given event type masked.

specific event types masked. The ability to destroy targets appears to be primarily dependent on the information contained in events “A” (within target's determine the aircraft's ability to evade danger. The event “D” (target destroyed) also appears to have some effect, although a less significant one. In this case, aircraft would continue trying to destroy targets that are already disabled. Again the 20% change on the “d ” event is most likely due to experimental error.

Figure 10 shows the amount of time required to complete the mission versus the percentage of specific event types masked. Loss of events “D” (target destroyed) and “e” (escape from danger) appear to prolong the mission. Loss of event “A” (within target's striking range) shortens the mission decisively. This is likely due to increased plane mortality. Changes in other events appear not to significantly affect the amount of time required to complete the mission.

These results provide many important pieces of information. They provide insight into the functioning of the DEDS controllers. They also indicate the pieces of information that are most essential for successful missions. Interestingly enough, while it is useful to know whether or not a target has been disabled it appears to be of secondary consequence. Most important are the events that allow the aircraft to avoid danger.

7. Hypothesis and scenario

Our primary hypothesis is that C^2 hierarchies should adapt to a changing battlespace. This hypothesis is built on two assumptions: (1) no single C^2 hierarchy is best for all air operations, and (2) uncertainty in the battlespace (*fog* and *friction*) makes it impossible to know the best hierarchy beforehand. To justify the first assumption, we provide the following explanation. In any operation some decisions, such as distribution of scarce resources, require central coordination. Other decisions, such as returning hostile fire, often require decision speed that only local reactions can provide. Successful C^2 requires both central coordination and local autonomy, with information flowing in all directions through the hierarchy [11].

The best hierarchy depends on technology, weather, and geography as well as friendly and enemy objectives, forces, positions, and strategies. The second assumption is justified by the omnipresence of uncertainty in battle [10]. It would be very unusual for friendly forces to know enemy objectives, forces, positions, and strategies with certainty before an operation. The amount of information available to a commander during an encounter increases as friendly forces engage the enemy. Similarly, *friction* makes it impossible to accurately predict the consequences of actions. As the amount of information increases, so does the commander's ability to create an appropriate C^2 structure.

In enemy airspace there is a target, which is to be attacked by bombers. The enemy has mobile Surface-to-Air Missiles (SAMs), fixed SAM's, and Anti-Aircraft Artillery guarding the target. Friendly forces send in Wild Weasel aircraft to find and destroy the enemy air defenses. All enemy air defenses within the corridor are to be disabled for the length of the mission. Figure 4 illustrates this scenario.

8.Coordination results

Figure 11 contains results from SEAD scenarios run using 40 Wild Weasel platforms to clear a corridor 200 km. wide by 300 long, containing 200 fixed Surface-Air Missile (SAM) sites. The simulation ran until the corridor was clear of all enemy SAM targets. The x axis groups simulations according to the amount of coordination used. The y axis is percentages. The top line gives the percent of area cleared and the bottom line provides the percent of planes surviving the mission. All numbers given are the average values obtained from multiple simulation runs.

The set of values to the left in figure 11 are obtained from runs with no centralized coordination. All forty planes are sent to concurrently clear a portion of the corridor. The middle set of runs had centralized coordination and perfect initial information. Twenty wild weasels were sent to concurrently clear nonintersecting portions of the corridor. As individual aircraft were destroyed, replacements were dispatched. The set of the runs on the right differed from the middle set in that the initial information was imperfect. Targets would be found by patrolling aircraft.

The initial conclusion drawn from these runs is coordination greatly improves the ability of the system to perform its mission. This is illustrated by the difference between the uncoordinated missions on the left and the other two. This is true even for a scenario that requires all platforms to act in a fairly independent manner.

A surprising result is the difference caused by partial observability of the central planner. As shown by the upper line the percent of the corridor kept clear decreases slightly, which is to be expected. The results shown by the lower line are unexpected. The percentage of planes surviving the mission increases. This can be explained by the fact that some targets are never found by the Wild Weasels during their reconnaissance flights. This means fewer targets are engaged, fewer attacks occur, and there are fewer casualties. Since more enemy SAMs would survive in the case with imperfect information, the corridor clearing would be less effective and endanger other aircraft.

9. Conclusion and further research

This paper has presented the conceptual underpinnings of using adaptive C^2 hierarchies for air operations. The concept is consistent with existing military literature in that *fog* and *friction* limit a commander's ability to foresee the future configuration of the battlespace. Recent work in distributed system analysis indicates that agent based simulations are the best available methodology for evaluating strategies in coordinating large systems of interacting decision makers.

We have presented related work from other researchers that explore multi-dimensional landscapes to find when centralized control is superior to decentralized control. We have also provided results from an initial in-house experiment that quantifies the utility of inter-agent coordination and the use of tactical intelligence in air combat.

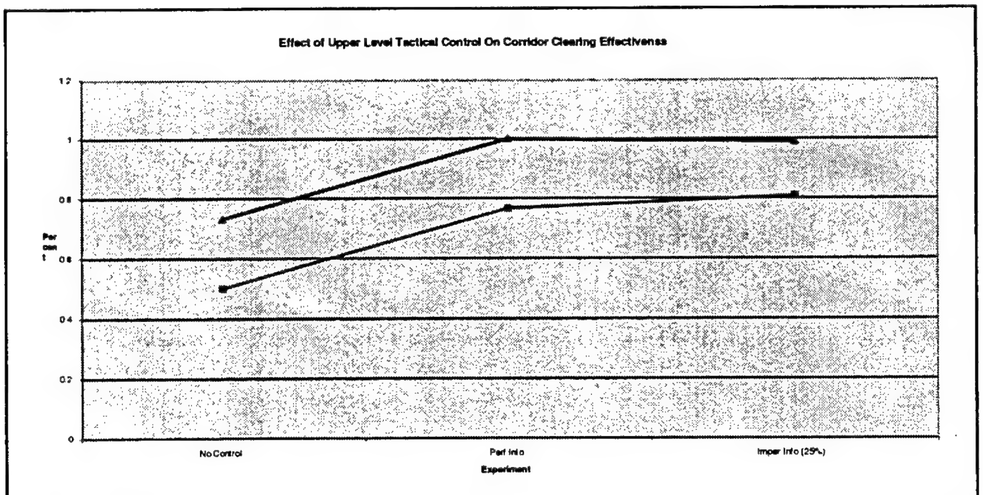


Figure 11. Effect of upper level control on mission effectiveness. The x axis is the amount of coordination. The y axis is percentages. The top line is the percent of the area cleared and the bottom line is the percent of aircraft surviving.

The final part of our paper has detailed experiments that quantify the ability of the DEDS controllers to perform corridor clearing scenarios. A number of experiments show their robustness to lack of information. We also analyzed the sensitivity of the controller to the loss of specific event types. The final set of experiments showed the utility of coordination among the controllers in performing corridor clearing.

Further research is being done in developing hierarchical controllers and the ability to adapt the control hierarchy to a given mission. Of special interest is the ability of the system to compress information for supervisory levels of the system, and sensitivity of the system to inconsistencies. The hierarchical control design methodology should be sufficient to avoid inconsistencies at different levels of aggregation, like those documented in [5].

10. Acknowledgements and disclaimers

This Effort is sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-99-1-0547 (JFACC). The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

11. References

[1] J. Bortz, M. Ownby, et al *JFACC Scenario Data, 2000.* http://jfacc.cgi.com/teams/AirOps/docs/Cyberland/go.htm

[2] A. K. Cebrowski and J. J. Garsta, "Network-Centric Warfare: Its Origin and Future," *Proceedings of the Naval Institute,* Jan 1998. http://www.usni.org/Proceedings/Articles98/PROcebrowski.htm.

[3] M.-H. Chang, and J. E. Harrington, “Centralization vs. Decentralization in a Multi-Unit Organization: a Computational Model of a Retail Chain as a Multi-Agent Adaptive System,” *Working Papers of the Santa Fe Institute, 1999.* http://www.santafe.edu/sfi/publications/Working-Papers/00-02-010.pdf

[4] T. Czerwinski, *Coping with the Bounds: Speculations on Nonlinearity in Military Affairs*, National Defense University, Washington, DC, 1998.

[5] P. K. Davis and J. H. Bigelow, *Experiments in Multiresolution Modeling MRM.* Report MRM-1004-DARPA. Rand Cooperation, Santa Monica, CA, 1998. http://www.rand.org/publications/MR/MR1004/

[6] K. Nagel, M. Rickert, P. M. Simon, and M. Pieck, “The dynamics of iterated transportation simulations,” *Working Papers of the Santa Fe Institute, 2000.* http://www.santafe.edu/sfi/publications/Working-Papers/00-02-012.pdf

[7] S. Phoha and R Brooks, "A Constructivist Theory of Distributed Intelligent Control of Complex Dynamic Systems," DARPA-JFACC *Symposium on Advances in Enterprise Control*, San Diego, CA, November 15-16 1999.

[8] S. Phoha and N. Gautam, and A. Horn, "Tactical Intelligence Tools for Distributed Intelligent Control of Air Operations," submitted to the 2nd DARPA-JFACC *Symposium on Advances in Enterprise Control*, Minneapolis, MN, July 10-11, 2000.

[9] Tzu, Sun , *L'Art de la Guerre*, Flamarrion, Paris, 1972.

[10] Carl von Clausewitz, *Vom Kriege*, Ferd. Duemmlers Verlag, Bonn, Germany, 1972.

[11] M. L. van Creveld, *Command in War,* Harvard University Press, Cambridge, MA, 1985.

[12] L. Wentz ed., *Lessons from Bosnia: The IFOR Experience.* National Defense University, Washington, DC, 1986.

[13] W. M. Wonham, *Notes on Control of Discrete Event Systems,* University of Toronto, April 1999.

[14] Wang Xi, A. Ray, S. Phoha, and W. Zhang, "Hierarchical Consistency of Supervisory Command and Control of Aircraft Operations, " submitted to the 2nd DARPA-JFACC *Symposium on Advances in Enterprise Control*, Minneapolis, MN, July 10-11, 2000.

[15] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control,* Prentice Hall PTR, Upper Saddle River, NJ, 1996.

Distinguishing Control and Plant Dynamics in Enterprise Modeling

H. Van Dyke Parunak[1], Sven Brueckner[1], John Sauter[1], and Robert S. Matthews[1]
ERIM Center for Electronic Commerce, PO Box 134001, Ann Arbor, MI 48113-4001 USA
{vparunak, sbrueckner, jsauter, rmatthews}@erim.org

Abstract

Two factors can confound the interpretation of an enterprise model. First, the dynamics of the control technology interact in complex ways with those of the plant, and engineers need to be able to distinguish these effects. Second, "mean field" approximations of the behavior of the system may be useful for qualitative examination of the dynamics, but can differ in surprising ways from the behavior that emerges from the interactions of discrete agents. This paper examines these effects in the context of a specific research project applying agent-based control to a military air operations scenario.

1. Introduction

Modern military operations can overwhelm a commander. The information available from satellite and other sensors floods conventional analysis methods. Enemy forces using advanced technology can hide or change location faster than conventional planning cycles can respond, and coordinating central orders across thousands of friendly resources can slow response even further. These features are prototypical of many modern enterprise control problems.

The ADAPTIV project (Adaptive control of Distributed Agents through Pheromone Techniques and Interactive Visualization) applies fine-grained agent techniques to the control of air resources charged with defending a friendly region from enemy attack. Intelligence on the location and strength of enemy ("Red") resources leads to the deposit of synthetic pheromones [1, 5, 7] in a spatial model of the battlespace. The propagation and evaporation of these pheromones model the uncertainty in the available intelligence. Friendly ("Blue") units then move in response to the flow field generated by these pheromones.

Initial experiments with these mechanisms are being conducted in a scenario dealing with the suppression of enemy air defenses, a task known as SEAD. Red is moving ground troops (GT), under cover of air defense units (AD), toward blue's territory. Blue has bombers (BMB) to hold back the ground troops, but these bombers are vulnerable to red AD. Blue also has fighters (SEAD) that can suppress the AD and thus protect BMB. The scenario takes the form of a strategy game, called "SEADy Storm," played on a hexagonally tiled field. Units of both sides can decide how to move over this field, and (when they find themselves in the same hexagon as one or more enemy units) whether to engage the adversary. Outcome rules determine how the strength of each unit changes as the result of an engagement.

Our initial experiments with pheromone mechanisms in this environment yield surprisingly complex variations in outcome as we vary the distribution of resources on each side. Some of this variation results from the simple strategies being executed by our agents, but much is due simply to the complexity of the game rules. To separate the two effects, we have done three successive layers of abstraction away from the initial experimental setting, first neutralizing the effect of Blue strategies, next removing the spatial structure of the game entirely, and finally abstracting away from the individual units with a mean field approximation to the combat rules.

We conducted these experiments to explore the potential of pheromone mechanisms, but this paper does not discuss these mechanisms. Our focus here is on two more general morals. First, experimental abstraction is a useful (even necessary) technique for understanding the relative effects of agents and environment (or to use the vocabulary of control theory, of the controller and the plant). Second, modeling technology can distort the picture in ways to which the analyst must be sensitive.

Section 2 describes in more detail the problem domain, the particular scenario we are exploring, and the initial results we obtained. Section 3 looks at three successive abstractions we applied to this scenario in an effort to distinguish environmental and agent dynamics. Section 4 discusses our experience and summarizes key insights

2. The SEADy Storm Experimental Context

First we summarize the structure and rules of our plant. Then we describe the behavior of our control agents, and show results from initial experiments.

2.1 The SEADy Storm Game

SEADy Storm [3] is a war game used to explore technologies for controlling air tasking orders. The battlespace is a hexagonal grid of sectors, each 50 km across (Figure 1). Friendly (Blue) forces defend a region in the lower left against invading Red forces that occupy most of the field. Red's playing pieces include ground troops (GT's) that are trying to invade the Blue territory, and air defense units (AD's, surface-to-air missile launchers) that protect the GT's from Blue attack. Blue has bombers (BMB's) that try to stop the GT's before they reach the blue territory, and fighters tasked with suppressing enemy air defenses (SEAD's).

Each class of unit has a set of commands from which it periodically chooses. In our implementation, ground-based units (GT and AD) choose a new command once every 12 hours, while air units (BMB and SEAD) choose once every five minutes. These times reflect the time it would take the resource to move across a sector. The commands fall into three categories (Table 1). GT cannot attack Blue forces, but can damage BMB's if they attack GT.

Blue can attack AD and GT when they are moving or attacking, and AD may attack any Blue forces that are not moving or waiting. Each unit has a strength that is reduced by combat. The strength of the battling units, together with nine outcome rules, determine the outcome of such engagements. Informally, the first five rules are:

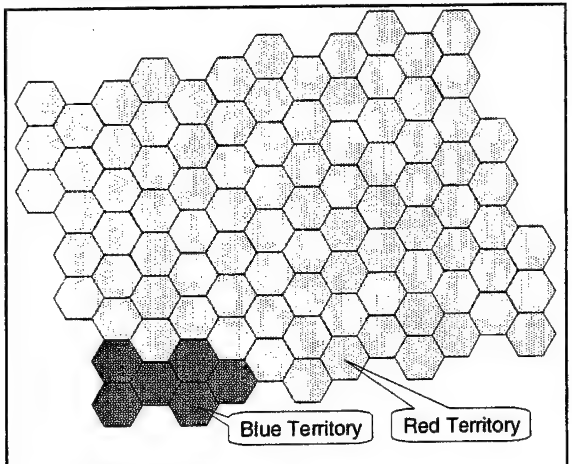


Fig. 1. The SEADy Storm Board.

Table 1. Unit Commands in SEADy Storm.

	Move	Attack	Wait
AD	Relocate	Fire (on any Blue aircraft)	Hide Deceive
GT	Advance		Hide
SEAD	NewSectors	AttackAD	Rest
BMB	NewSectors	AttackAD AttackGT	Rest

1. Fatigue: The farther Blue flies, the weaker it gets.
2. Deception: Blue strength decreases for each AD in the same sector that is hiding.
3. Maintenance: Blue strength decreases if units do not rest on a regular basis.
4. Surprise: The effectiveness of an AD attack doubles the first shift after the unit does something other than attack.
5. Cover: BMB losses are greater if the BMB is not accompanied by enough SEAD.

Rules 6-9 specify the percentage losses in strength for the units engaged in a battle, on the basis of the command they are currently executing. For example, Rule 9, in full detail, states: "If BMB does "AttackGT" and GT does "Advance": a GT unit loses 10% for each BMB unit per shift; a BMB unit loses 2% per GT unit per shift."

2.2 The ADAPTIV Mechanisms

ADAPTIV uses pheromone techniques to control Blue operations in such a scenario. Intelligence reports on Red locations deposit synthetic pheromones in a spatial model corresponding to the hexagonal grid. The pheromone infrastructure *evaporates* pheromones to model the decreased value of stale information, *propagates* pheromones from one sector to another to pass information to nearby resources, and *aggregates* deposits from subsequent reports to highlight new information. The experiments reported in this paper manipulate a package of BMB and SEAD as a unit.

Each class of unit must wait (5 minutes for a BMB-SEAD package, 12 hours for AD and GT) after executing one command before choosing another. When a unit is eligible for a new command, it selects with equal probability from its possible commands. If it selects a movement command, the movement depends on the unit in question and the pheromones it senses in the six adjacent sectors. A unit "follows" the pheromone field using a roulette wheel weighted by the strength of the appropriate pheromone in the neighboring sectors. In this experiment,

- the SEAD-BMB package follows GT pheromones,
- AD units follow a product of BMB and GT pheromones, thus seeking out BMB's that are threatening GT's, and

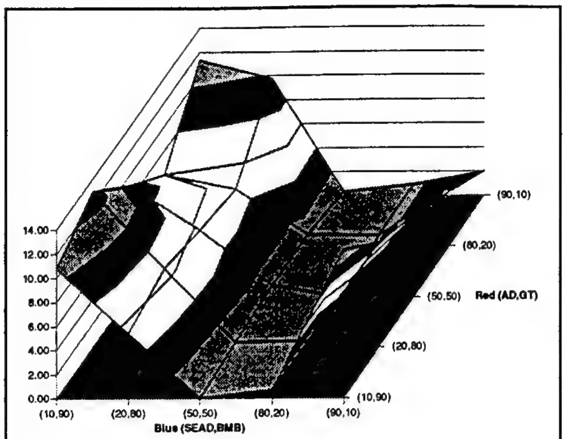


Fig. 2. Red Strength in Blue as a function of force composition (EXP)

- GT units move randomly, with higher weights given to south-westerly neighbors.

An experiment runs for 1200 simulated hours (about 7 minutes on a 500 MHz Wintel). At the end of the experiment we calculate the total Red strength that has reached Blue territory ("Red in Blue" or "RinB") and the surviving percentages of each class of unit. We run each configuration of parameters eleven times with different random seeds, and report medians for each configuration.

2.3 EXP: Experimental Results

The primary parameter explored in the experiments reported here is the proportion of SEAD in the Blue military, and of AD in the Red military. Each side began with a 100 units, each with unit strength, and 10%, 20%, 50%, 80%, or 90% of SEAD or AD. The uneven spacing reflects a basic statistical intuition that interesting behaviors tend to be concentrated toward the extremes of percentage-based parameters. In current military doctrine, 50% is an upper limit on both AD and SEAD. We explore higher values simply to characterize the behavioral space of our mechanisms.)

The central outcome is total Red strength in Blue territory at the end of the run (Figure 2). The landscape shows several interesting features, including

- a "valley" of Blue dominance for all Red ratios when Blue SEAD is between 50% and 80%, with slightly increasing Red success as the AD proportion increases;
- clear Red dominance for lower SEAD/BMB ratios, decreasing as SEAD increases;
- a surprising increase in Red success for the high SEAD and low AD levels.

Figure 3 shows the percentages of each class of unit surviving at the end of the run. The AD, GT, and BMB plots reflect the main features of the topology. The increase in Red effectiveness for high SEAD appears to be due to a drop in BMB survival in this region, a surprising effect since BMB's have strong SEAD protection here.

The overall system shows interesting and non-trivial dynamics, with two sources: the pheromone-based movement of the resources, and the outcome rules that define the scenario. Early study, for example, showed us that the Red superiority for low SEAD ratios is directly related to Rule 5, which places a particularly heavy penalty on Blue packages that do not have at least one SEAD for every two BMB's. This rule induces a threshold nonlinearity at SEAD/BMB 33/67, which marks the edge of the Blue valley in other runs (not shown here) that explore the parameter space in more detail. However good Blue's

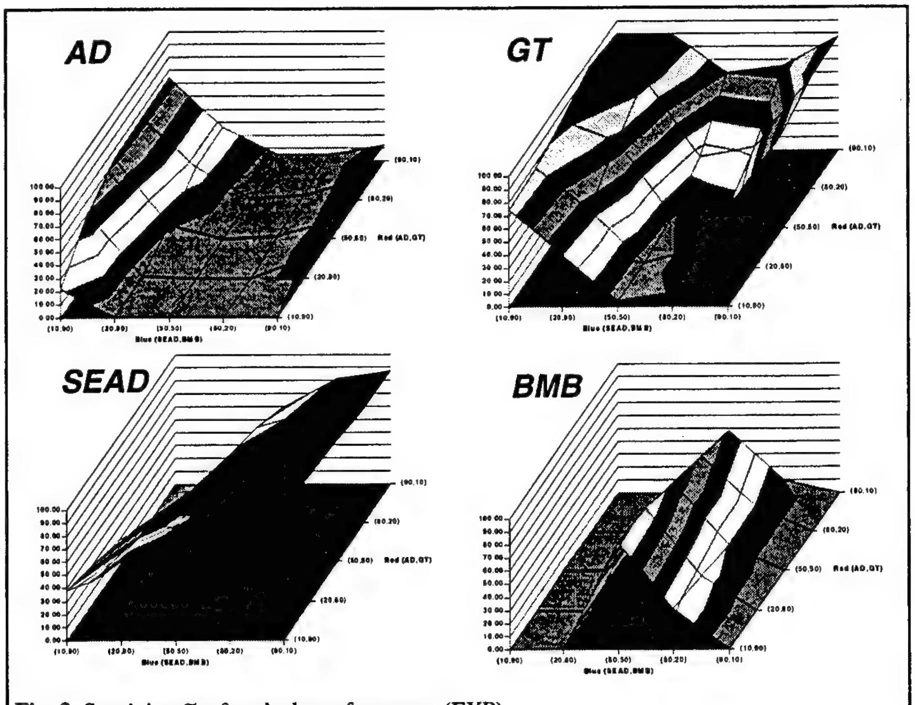


Fig. 3. Surviving % of each class of resource (EXP)

pheromone algorithms are at finding and targeting Red troops, Rule 5 will impose a performance cliff along this parameter.

3 Successive Abstractions

To understand the contribution of our control mechanisms, we must distinguish their dynamics from those of the plant with which they interact. We abstract away successive details of our mechanisms and compare the resulting system behaviors with those of the full system. First, we performed a mean field abstraction (ABS3) that removes the effects of Blue strategy, spatial distribution, and the distinction among individual agents. This abstraction behaved sufficiently differently from EXP that we examined two intermediate abstractions, one removing only blue strategy (ABS1), the other removing blue strategy and spatial distribution (ABS2). We present the abstractions in logical sequence rather than in chronological order.

3.1 ABS1: Ignoring Blue Strategy

Blue units find Red targets by climbing pheromone gradients. A logical abstraction is to "cut off their noses," moving Blue randomly rather than in response to pheromone signals. Figure 4 shows the strength of Red in Blue at the end of the game under these conditions. The landscape has the same general features as Figure 2.

We can compare the two by subtracting at each point the Red in Blue strength when Blue moves randomly from that when Blue follows pheromones, as in Figure 5. Because Blue seeks to keep Red out of Blue territory, differences less than 0 represent a net contribution of the Blue mechanisms. Figure 5 shows that our mechanisms are generally effective, with the greatest benefit at 10% SEAD, 50% AD. There are two exceptional regions where random wandering outperforms pheromones.

The first is when Red AD is above 50%. In this region, BMB survival is worse in EXP than in ABS1 (Figure 6), leading us to hypothesize two possible causes for the difference. It may result from the movement rule we have assigned to AD, to follow GT weighted by BMB pheromones. If BMB movement is regular (guided by slow-moving GT), AD's can position themselves more effectively. When BMB move randomly, AD's have a harder time positioning themselves for maximum impact. Another explanation recognizes that with high AD coverage around GT, a frontal attack of BMB on GT takes them into the most deadly opposition, while random movement will sometimes encounter weaker Red units that they can more effectively pick off. Distinguishing these alternative effects requires further experiments.

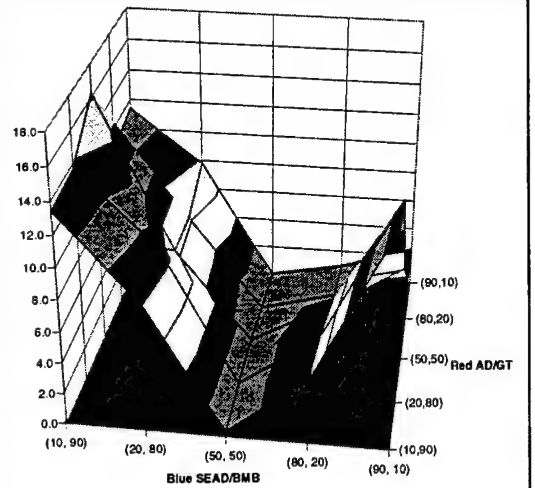


Fig. 4. Red Strength in Blue, without pheromone effects (ABS1)

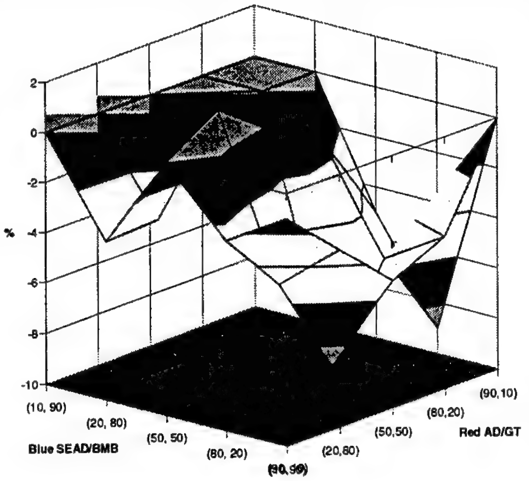


Fig. 5. BMB survival (EXP – ABS1)

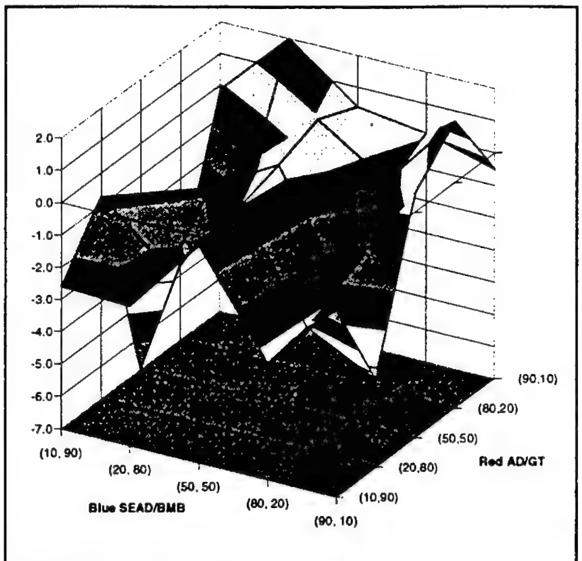


Fig. 6. Red in Blue (EXP – ABS1)

Lesson: Within the same problem domain, parametric differences can lead to a very different interaction between the agents and their environment, and agents need to be designed to take these differences into account in determining their behavior.

Second, even with low Red AD, the pheromone mechanisms make little or no contribution at 50% SEAD. This effect is probably due to a lack of opportunity. The valley around 50% SEAD and low AD is intrinsically so favorable to Blue that Blue's strategy makes little difference.

Lesson: Success may result from luck rather than intelligence. Environmental dynamics can be so strong that agent intelligence makes little or no difference.

Figure 7 shows the difference in AD survival between EXP and ABS1. As we have seen, pheromones make little difference toward the low-AD end of the 50% SEAD valley, and help Blue at 10% SEAD, 50% AD. But they are a detriment around the edges of the valley. The right-hand ridge reflects the puzzle we have already seen in Figure 2: why should higher SEAD strength lead to better Red success? Figure 7 shows that this anomaly is reflected in AD survival. Initially, this circumstance is even more puzzling than high Red in Blue in this region. Why should higher SEAD help AD survival? By focusing our attention on the AD units, this plot leads us to the answer, a complex chain of interlocking events.

1. Rule 9 specifies that when BMB attacks GT, the losses on each side depend on the ratio of GT to BMB. When SEAD percentage is high, a package contains fewer BMB's, and the GT/BMB ratio is higher. For very high SEAD levels, BMB is at a disadvantage in an encounter with GT, accounting for the lower survival of BMB at high SEAD.

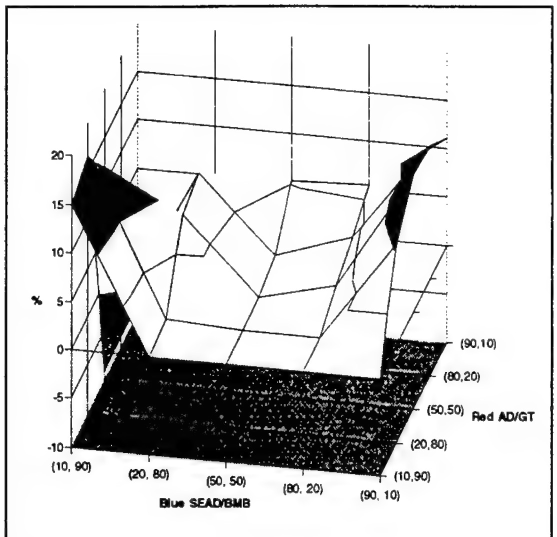


Fig. 7. AD survival (EXP – ABS1)

2. AD's are attracted by GT pheromones, weighted by BMB pheromones. The BMB population falls off at high SEAD levels, both intrinsically and because of the dynamic in the previous point. Thus AD movement becomes more random, and AD's are less likely to be found in the close vicinity of GT.
3. SEAD's travel with BMB's in Blue packages, which are attracted by GT. As AD's wander more, they are less likely to be near GT's, thus less likely to encounter SEAD's, and their survival increases.
4. Meanwhile (returning to the right-hand flap in Figure 2), the decreased population of BMB's leaves GT free to invade Blue territory.

Lesson: Even simple rules interact in complex and unanticipated ways. Careful analysis is necessary to understand the implications of these interactions for the design of individual agent behaviors.

3.2 ABS2: Ignoring Spatial Distribution

ABS1 shows us the effect of turning off Blue's pheromone mechanisms, but Red's deliberate movement is another layer that we must peel away from the system behavior to understand the impact of the outcome rules. One way to remove the effect of these mechanisms would be to randomize Red's movement as well as Blue's, but this would still leave a dependency on Red's initial spatial distribution. Alternatively, we can remove space entirely, so that all units occupy a single sector.

Historically, we analyzed ABS2 in order to validate the mean field approach of ABS3, which is intrinsically non-spatial, so ABS2 pursues the single-sector approach. This abstraction changes how Red and Blue agents encounter one another. In the spatial model, agents interact when they find themselves in a common sector. As a result, agent movement (whether random or purposeful) induces a distribution on how many agents can be engaged at a given time step. For example, Blue in a sector with no Red forces can neither cause nor receive battle damage. Under such circumstances, an "attack" command is effectively a no-op. When we place all resources in the same sector, we need another way to model how many resources will be engaged. Thus we define the proportion of each type of unit that will execute each eligible command at each time step. In the results reported here, we assign the following parameters (parameter set a), based on the results from this same set in ABS3:

- AD: 0% Hide, 0% Deceive, 10% Fire, 90% Relocate (to the same sector)
- GT: 80% Hide, 20% Advance (to the same sector)
- SEAD: 10% Rest, 90% Attack AD, 0% New Sector
- BMB: 60% Rest, 20% Attack AD, 20% Attack GT , 0% New Sector

For example, at a given time step, a randomly selected 80% of the GT's will Hide, while the others will Advance (thus being vulnerable to attack).

Figure 8 summarizes some results from these parameters, compared with EXP and ABS1. These plots show several interesting features.

The topography in ABS2-a is shifted toward lower SEAD percentages, relative to that in EXP and ABS1. The valley in surviving GT and the peak in surviving BMB now fall between 20% and 50% SEAD, rather than at or beyond 50% SEAD as before. The location of the valley reflects the penalty imposed by Rule 5 when the ratio of SEAD to BMB falls below 1/2. In ABS1, SEAD and BMB are packaged based on the overall percentage of SEAD, which is thus involved in any combat. In ABS2-a, the proportion of SEAD and BMB in a conflict depends not only on the overall percentage of SEAD, but also on the number of each that is resting and out of action on a given cycle. The command percentages in Figure 8 make 90% of SEAD available to attack on any given cycle, but only 40% of BMB. Thus the effective SEAD percentage is more than twice the overall SEAD percentage, and ABS2-a shows the same effect at 20% SEAD that EXP and ABS1 show at 50% SEAD.

The obvious fix is to fit the command percentages more carefully to the distribution induced by movement in the spatially distributed case. Such a fit is more easily requested than delivered. The complexity of various movement rules makes an analytical derivation intractable. The desired distribution is probably not even stationary, since

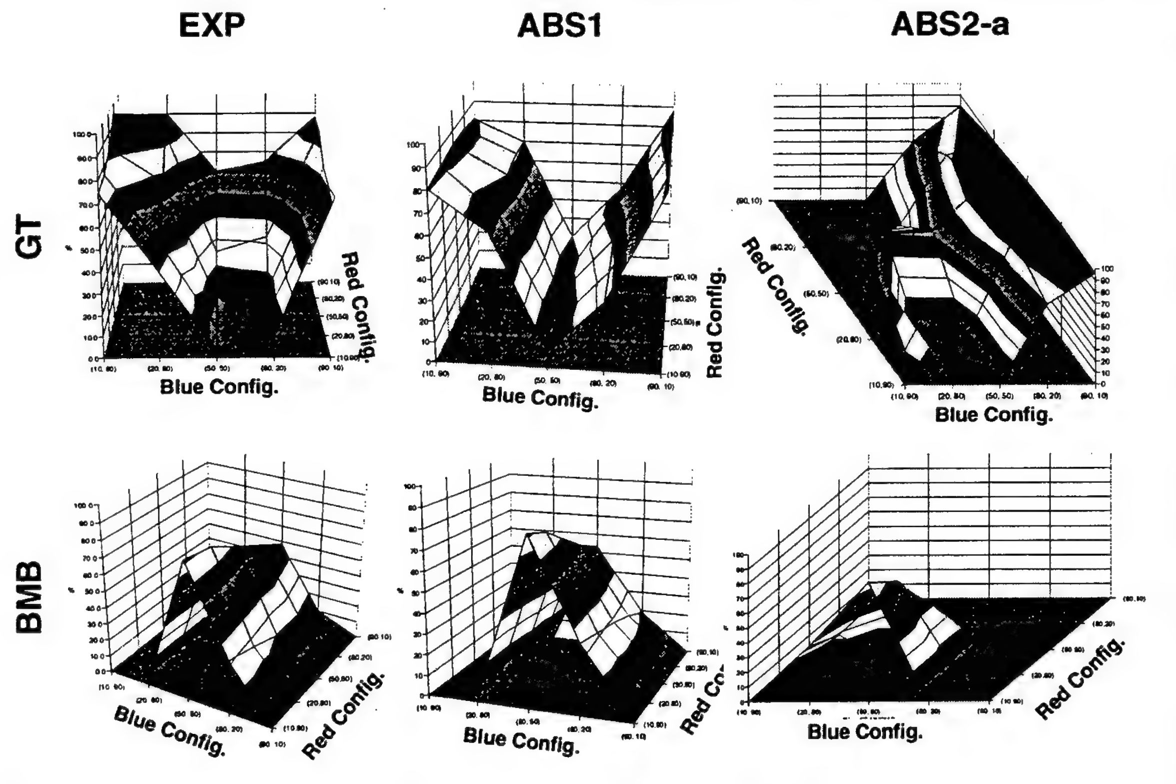


Fig. 8. Comparison of surviving GT and BMB percentages for EXP, ABS1, and ABS2-a.

population changes over a run will change the probability that two units will encounter one another. One could use a visual or statistical match between performance landscapes such as those in Figure 8 to determine command percentages experimentally. More fundamentally, these observations call into question the validity of aspatial models of spatially distributed problems.

Lesson: Space is not just a neutral medium in which agents interact. It plays an active and complex role in their interactions, a role that is difficult if not impossible to capture without modeling space directly.

ABS2-a also differs from EXP and ABS1 in the nonlinearity of GT's dependence on AD percentage. In the previous experiments, the valley rises monotonically as AD increases. In ABS2-a, this rise peaks at 50% AD, then falls sharply for higher AD percentages. At this point, we do not have a detailed explanation for this feature. It is unlikely that we will devote considerable effort to understanding it, since the basic lesson of ABS2-a is that removing spatial distribution entirely from the model is not a fruitful approach to our objective of factoring agent effects from environment effects.

3.3 ABS3: Ignoring Unit Identity

Execution of agent-based models can be very time-consuming (in our case, requiring 7 minutes to simulate 1200 hours). Eleven replications at each of 5x5 = 25 Red/Blue configurations require over 32 hours to yield experimental results. A parallel equation-based model (ABS3) requires about 1.5 seconds to simulate 1200 hours. Significant differences between agent-based and equation-based models [8, 10] make the agent-based model the gold standard for evaluating our pheromone methods, but rapid surveys of parameter space with an equation-based model might guide more meticulous (and time-consuming) verification using the agent-based model.

ABS3 uses a population-based modeling approach where the aggregated strength of all units of one type (AD, GT, SEAD, BMB) is represented in the size of one distinct population. The size of a population changes over time. The change is determined by the portion of each population that engages in combat in each discrete time step and by the losses inflicted in these combats.

We represent the population dynamics in a set of difference equations that capture both, the combat composition and the outcome rules. For example, in the GT population the population size is reduced by in every step by

$$\Delta GT = g(GT, Advance, BMB, AttackGT) * BMB * \frac{c(t, BMB, AttackGT)}{c(t, GT, Advance)}$$

where

- $g(X,a,Y,b)$ represents the percent losses of a group of units of type X that executes the command a when it engages a group of units of type Y that executes the command b at a time step. Losses are specified in outcome rules five to nine.
- $c(t,X,a)$ specifies the combat composition, and represents the percentage of population X that executes the command a at time t. The ABS3 experiments all assume a constant combat composition: $c(t1,X,a) = c(t2,X,a)$ for all pairs $(t1, t2)$.

ABS3 initializes the four populations to represent the initial strength of the combatants. Then, for a specified number of time steps, it

- computes the decrease in the population size for each population;
- limits the computed losses to the portion of each population that is actually engaged in combat (given by: $X*c(t,X,a)$, where a is an attack command); and
- applies the losses.

The number of time steps is the same as the number of calls in a comparable run of EXP to the function resolving combat situations.

With ABS3, we were able to generate landscapes that matched those from EXP and ABS1 qualitatively, but not in detail. The differences might be explained either by the move from an agent model to an equation one, or by the collapse of space. We constructed ABS2 in an effort to tease apart those rival effects.

As we have seen, collapsing space does make a difference, due largely to the necessity to capture in static command probabilities the distribution of activities induced by agent encounters as they move through space. Comparison of ABS2 with ABS3 shows that the move from agents to equations has other effects as well.

Figure 9 compares three pairs of surviving GT and BMB landscapes. ABS3-a uses parameter set a (defined in Section 3.2), and shows that even with a space-free model, command percentages can be tuned to produce landscapes similar to those in ABS1 (or, for that matter, EXP; compare Figure 3). We ran ABS2-a with these same percentages to test whether the shift to an equation-based model makes a difference. Figure 9 shows that it does. The percentages that produce realistic landscapes in ABS3-a lead to the anomalies we have already discussed in ABS2-a. The third column in Figure 9 shows landscapes in ABS3-b, with different command parameters chosen to make these landscapes resemble ABS2. Parameter set b is AD = {0.0,0.0,0.5,0.5}; GT

= {0.8,0.2}; SEAD = {0.45,0.55,0.0}; BMB = {0.2,0.4,0.4,0.0}.

Thus ABS3 can show us the existence of interesting non-trivial performance landscapes, but for a given set of parameters, it cannot reliably tell us either the location or the topology of their features. The salient difference between ABS2 and ABS3 is that ABS2 retains distinct agents, while ABS3 represents only the aggregate strength of the entire population of agents of a given type (thus, a single strength for each of AD, GT, SEAD, and BMB). The strength of individual agents in ABS2 evolves from the engagements in which each agent is involved, and thus summarizes that agent's history. ABS3 loses this history. The simulation logs show different evolution of the total strength over time in the two cases, leading to the different final outcomes reflected in Figure 9. The effect is closely related to the sensitive dependence of nonlinear systems on initial conditions. Once individual agents in ABS2 come to differ slightly in their strengths, their subsequent evolution can diverge greatly, leading to changes in the outcome of subsequent combats. ABS3 cannot track these different histories, and so is insensitive to their results.

Lesson: Like spatial distribution, ontological distribution (distributing processing over independent interacting processes) makes a substantive and non-intuitive difference to the outcome of a model. Whether or not a mean-field model works in a given situation is an empirical question. Such models must be carefully validated against agent-based models before being trusted.

4 Discussion and Summary

The world is too complicated a place to understand as it exists. Science seeks to understand it by abstracting away details to leave a simplified system, and manipulating that system with a modeling technology (such as mathematical analysis or simulation). The validity of this process requires that neither the abstraction nor the modeling representation substantially change the behavior of the system. As our experience shows, both of these requirements are easily compromised.

First, abstraction requires that the system under study be cleanly separable from the environment in which it is usually embedded. Designers of agent-based systems typically pay much more attention to the agents than to the environment. The complex interactions discussed in this paper show that the environment deserves more attention. Our observations support researchers in embodied cognitive science [9] who argue that the agent and its environment must be designed together. The behavior of interest is that of the whole system, and only by

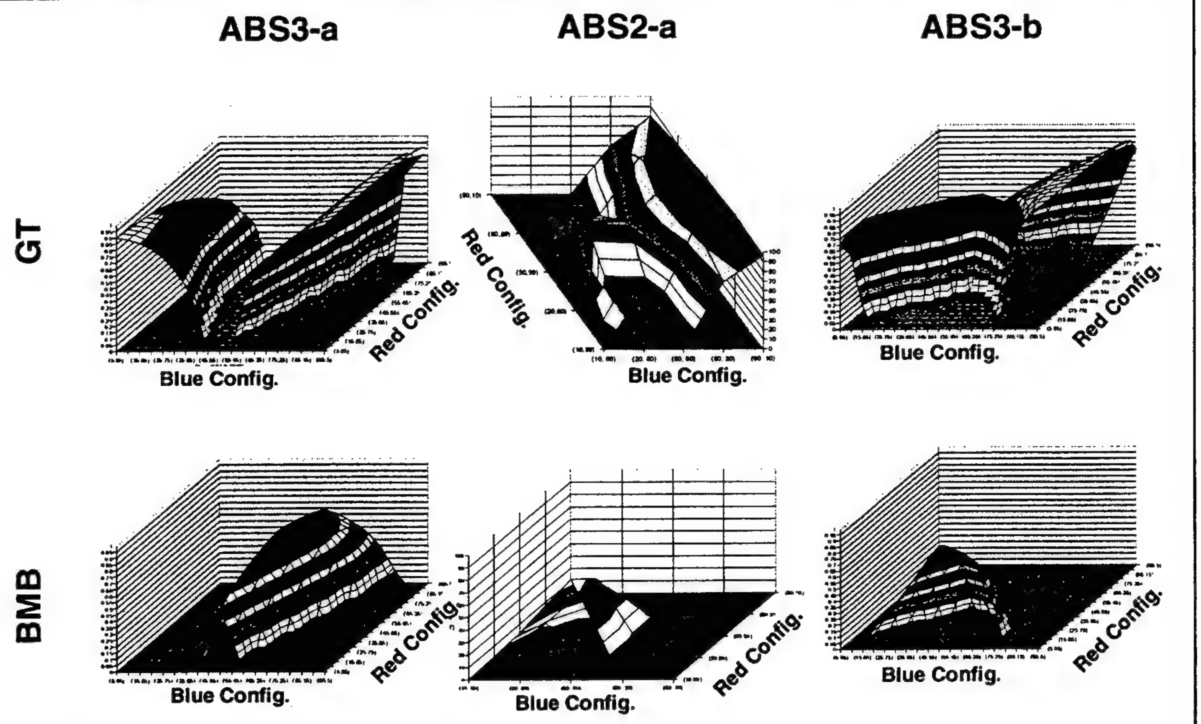


Fig. 9. Abstracting away from individual agents leads to further changes in behavior.

considering the environment with the agent can we reliably design systems that do what we wish. The abstraction process exemplified in this paper is a methodological tool that can make us aware of how our systems interact with their environments.

Second, modeling technologies are not content-neutral. Their mechanisms can introduce artifacts determined more by the modeling technology than by the system being modeled. Earlier researchers have pointed out such effects within agent-based models, based on differences between synchronous and asynchronous execution [2, 4]. Our results in this paper reinforce our earlier observations [8] about the loss of ontological distribution in an equation-based model.

Our conclusion is cautionary, not fatalistic. We do not reject system modeling and simulation as impossible. In fact, it is unavoidable in engineering control systems for complex enterprises, due to the analytical intractability of typical systems and their strongly nonlinear behavior [6]. We do warn that simulation is nontrivial. It forms a new way of doing science, alongside physical experimentation and mathematical analysis. These classical modes have evolved methodological guidelines for reliable results. Effective simulation science requires the development of similar guidelines, and the particular potential of agent-based modeling suggests that agent researchers should be in the forefront of developing this methodology. This paper is a step in this direction.

Acknowledgements

Olga Gilmore and Murali Nandula of the ERIM CEC staff contributed significantly toward the experimentation reported in this paper. This work is supported in part by the DARPA JFACC program under contract F30602-99-C-0202 to ERIM CEC. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government.

References

[1] S. Brueckner. *Return from the Ant: Synthetic Ecosystems for Manufacturing Control*. Ph.D. Thesis at Humboldt University Berlin, Department of Computer Science, 2000.

[2] B. A. Huberman and N. S. Glance. Evolutionary Games and Computer Simulations. *Proceedings of the National Academy of Science USA*, 90(August):7716-7718, 1993.

[3] A. Kott. SEADy Storm. 2000. Web Page, http://www.cgi.com/web2/govt/seadystorm.doc.

[4] S. E. Page. On Incentives and Updating in Agent Based Models. *Computational Economics*, 10:67-87, 1997.

[5] H. V. D. Parunak. 'Go to the Ant': Engineering Principles from Natural Agent Systems. *Annals of Operations Research*, 75:69-101, 1997. Available at http://www.erim.org/~van/gotoant.pdf.

[6] H. V. D. Parunak. From Chaos to Commerce: Practical Issues and Research Opportunities in the Nonlinear Dynamics of Decentralized Manufacturing Systems. In *Proceedings of Second International Workshop on Intelligent Manufacturing Systems*, pages k15-k25, K.U.Leuven, 1999.

[7] H. V. D. Parunak and S. Brueckner. Ant-Like Missionaries and Cannibals: Synthetic Pheromones for Distributed Motion Control. In *Proceedings of Fourth International Conference on Autonomous Agents (Agents 2000)*, 2000. Available at www.erim.org/~vparunak/MissCann.pdf.

[8] H. V. D. Parunak, R. Savit, and R. L. Riolo. Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide. In *Proceedings of Workshop on Multi-agent systems and Agent-based Simulation (MABS'98)*, pages 10-25, Springer, 1998. Available at http://www.erim.org/~van/mabs98.pdf.

[9] R. Pfeifer and C. Scheier. *Understanding Intelligence*. Cambridge, MA, MIT Press, 1999.

[10] W. G. Wilson. Resolving Discrepancies between Deterministic Population Models and Individual-Based Simulations. *American Naturalist*, 151(2):116-134, 1998.

Multiple Pheromones for Improved Guidance

Sven Brueckner[1] and H. Van Dyke Parunak
ERIM Center for Electronic Commerce
PO Box 134001
Ann Arbor, MI 48113-4001 USA
{sbrueckner, vparunak}@erim.org

Abstract

Synthetic pheromone systems offer great potential for spatial coordination in multi-agent systems. Initial experiments with such a system applied to the control of air operations has identified the concept of local guidance that is critical to designing such a system, and that can be supported by using multiple pheromones with differing characteristics. This paper reviews the basic mechanisms of synthetic pheromones, describes local guidance and how multiple pheromones support it, and outlines design methods to guide system designers in exploiting this mechanism.

1 Introduction

The synthetic ecosystems approach applies basic principles of natural agent systems to the design of artificial multi-agent systems ([4],[1]). Natural agent systems, like social insect colonies or market economies, express system-level features that make them interesting blueprints for industrial applications. Made up of a large number of simple, locally interacting individuals, these systems are flexible to changing conditions, robust to component failure, scalable in size, adaptive to new environments, and intuitive in their structure.

In natural agent systems, large numbers of individuals coordinate their activities in the fulfillment of tasks in stigmergetic interactions through the environment ([3]). The pheromone infrastructure, proposed in [2], enhances the execution infrastructure of our software agents, providing them with an active environment where they may share information. The pheromone infrastructure introduces a spatial structure to the system in which the agents may deposit synthetic pheromones at discrete locations (places) and perceive concentrations of such pheromones.

The internal operation of the pheromone infrastructure aggregates and propagates pheromone deposits by the agents. At the same time, local pheromone concentrations are reduced in strength automatically by the pheromone infrastructure's evaporation mechanism. There are three general parameters specifying a pheromone in the infrastructure: the pheromone's evaporation factor, propagation factor, and threshold. The evaporation factor determines the rate of the decay of the local strength of a pheromone over time. The propagation factor influences the strength with which a pheromone deposit event to a place is propagated to the neighboring places. The threshold is the strength below which the pheromone is ignored by the pheromone infrastructure. The performance of a pheromone-based coordination mechanism in a specific application depends on these three parameters.

Our paper reports a pheromone-based coordination mechanism of agents on a hexagonal grid. Agents of two species live in places on the grid: pumps and walkers. Pumps regularly deposit pheromones at their current place. Potentially, they are able to move independently over the grid, but in this paper we consider static pumps only. The walkers seek to occupy the same places as the pumps, but do not perceive them directly or know the purpose of their movements. Walkers are only permitted to sample pheromone concentrations at their current place and their immediate neighbors. They may not even communicate directly among themselves. This specific instance of the spatial coordination problem arose in the JFACC ADAPTIV project ([5]) in the tasking of air-combats, where a population of Bomber-agents has to find agents of Air-Defense or Ground-Troop units. Similar scenarios occur in civic domains like traffic coordination or manufacturing control.

Section 2 of this paper reviews pheromone mechanisms and defines some formal concepts. Section 3 presents some experimental results from ADAPTIV that focused our attention on two important characteristics of pheromone-based guidance and led us to formulate a preliminary hypothesis. Section 4 reports an analysis that challenges this hypothesis, but suggests an alternative, and describes a confirmatory experiment. Section 5 presents design recommendations for synthetic pheromone systems based on this understanding and Section 6 verifies the predicted performance improvement in a small experiment. We conclude in Section 7.

[1] Corresponding Author, Phone: (734) 623 2529 Fax: (734) 623 2501

2 Walking on Pheromones

A pheromone system embodies two sets of dynamics: those of the pheromones themselves, and those of the walkers, which move in response to the pheromones.

2.1 Pheromone Dynamics

Consider a stationary pump that deposits a fixed amount *A* of a pheromone at a fixed rate of one deposit every *T* unit time. The long-term behavior of the resulting pheromone field surrounding the pump depends on three parameters: the evaporation factor, the propagation factor, and the threshold of the pheromone.

Evaporation and propagation are inspired directly by physical processes in the real world, where they both result from Brownian movement of pheromone molecules. Evaporation models the removal of molecules from a place by Brownian motion. Some molecules settle on nearby ground where they may be sensed by ants. The propagation of deposit events in the pheromone infrastructure reflects this process.

Unlike evaporation and propagation, the threshold is a concession to the exigencies of a computational model. Physical processes in nature have no problem in bouncing a pheromone molecule anywhere on earth from its point of original deposit, but we model the passage of a pheromone from one place to another as a message in an object-oriented program, and the volume of messages would explode if we continued to pass pheromones whose strengths have decayed so far that they have no further practical effect. So, when a place receives a pheromone deposit below the threshold, it changes the local pheromone concentration, but does not propagate it farther.

If a place propagates a pheromone deposit to its direct neighbors, it determines the new deposit strength for each neighbor as the product of the original deposit strength and the propagation factor divided by the overall number of direct neighbors. The strength of a deposit weakens with every propagation step, because the propagation factor is required to be smaller than one.

A deposit at a place changes the local concentration of the pheromone by the strength of the deposit. Without any deposits, the local concentration of the pheromone is continuously reduced over time. The remaining concentration after one unit time is the product of the previous concentration and the evaporation factor of the pheromone.

A more detailed discussion of the pheromone dynamics in the generic pheromone infrastructure is presented in [2] and a forthcoming ERIM technical report.

2.2 Walker Behavior

All walkers move on the hexagonal grid in discrete steps. At a relocation moment t and located at an arbitrary place p, a walker selects its next location probabilistically from the set (*C*(*p*)) of currently available options. *C*(*p*) comprises the current place *p* and all of *p*'s direct neighbors. On the hexagonal grid away from the outside borders, a walker always has seven places ($C(p)=p_1,\ldots,p_7$) from which to choose. The following discussion assumes that the grid is sufficiently large to ignore the special case of places located at the grid's border.

The walker determines the selection probability of the places in two steps. First, it samples the concentration of the pheromone (s_i) at each place (p_i). In the second step, the walker determines the relative attraction (f_i) of a place as its local pheromone concentration normalized by the overall concentration of all places ($f_i = s_i / \sum_{p_j \in C(p)} s_j$).

As a result, the walker has assigned each place a number between zero and one, which add up to one across all seven places. The relative attraction is the probability of a place to be selected. The walker chooses its next place using a roulette wheel weighted according to these probabilities. The *local guidance at place p* available to the walker is $g(p) = \underset{p_i \in C(p)}{Max}(f_i) - 1/|C(p)|$, and ranges from 0 (if the pheromone has the same strength in all seven places) to *1-1/|C(p)|* (if only one place has a pheromone concentration larger than zero).

The pheromone-biased selection mechanism realizes a probabilistic climbing of the spatial gradient of the pheromone field. The stronger the gradient of the pheromone concentration is, the higher is the probability of the walker to follow the gradient.

3 Initial Experimental Results

We performed a series of experiments to study the effect of the evaporation and propagation parameters on the local guidance. Figure 1 shows the experimental setup, with 50 pumps distributed randomly in the western part of a 10x10 hexagonal grid.

```
              Pump Population

  0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0
  0   0   0   0   0   0   0   0   0   0
0   0   0   0   2   0   0   0   0   0
  0   0   0   0   2   0   0   0   0   0
0   0   3   3   0   0   0   0   0   0
  0   0   0   0   0   0   0   0   0   0
0   0   3   2   0   0   0   0   0   0
  0   2   0   0   0   0   0   0   0   0
0   0   0   0   0   3   0   0   0   0
  0   0   5   0   0   0   0   0   0   0
0   0   0   0   2   0   0   0   0   0
  0   0   0   0   0   3   0   0   0   0
0   0   0   0   0   0   0   0   0   0
  0   5   0   2   0   2   0   0   0   0
3   0   0   2   0   0   0   0   0   0
  0   0   3   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0
  0   0   0   0   0   3   0   0   0   0
0   0   0   0   0   0   0   0   0   0
```

Figure 1. Fifty pumps on a hexagonal grid

Figure 2 shows a contour plot of the local guidance that results when the pumps generate pheromone deposits with an evaporation factor *E=1/10* and a propagation factor *F=1/10*. The guidance is zero along the eastern portion of the grid, representing a wide valley across which the pheromone cannot propagate and in which walkers receive no guidance. In the western portion of the grid, where the pumps are located, the guidance is highly variable and frequently becomes quite high. If a walker in this region senses low guidance, a short random walk will bring it to a place with high guidance. Thus in this configuration, walkers have difficulty finding a target-rich area, but once within it, can home in quickly on individual targets.

$E=\frac{1}{10}\quad F=\frac{1}{10}\quad S=\frac{1}{1000}$

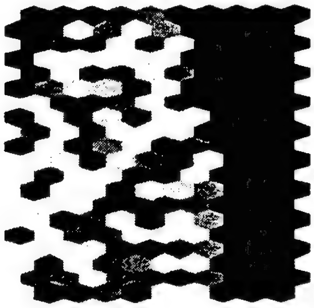


Figure 2. Local guidance for F=1/10

Experimentation shows that this picture does not substantially change with *E*. However, it is quite sensitive to *F*, as Figure 3 shows for *F=9/10*. Now the valley is considerably narrower, but the western area is dominated by a crater in which guidance is quite low. In this configuration, walkers can more easily locate the target-rich area, but then have difficulty homing in on individual targets.

$E=\frac{1}{10}\quad F=\frac{9}{10}\quad S=\frac{1}{1000}$

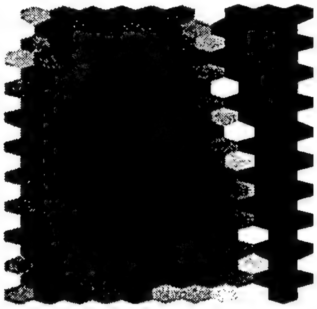


Figure 3. Local guidance for F=9/10

These observations led to the hypothesis that a pheromone has an "applicability range," an area on the hexagonal grid where the information in the local pheromone pattern is high enough (by some threshold value) to provide guidance to an agent. The applicability range forms an annulus around the pump (Fig. 4). A walker that selects its new location following the pheromone's gradient when it is too close to the pump or too far away, effectively moves at random.

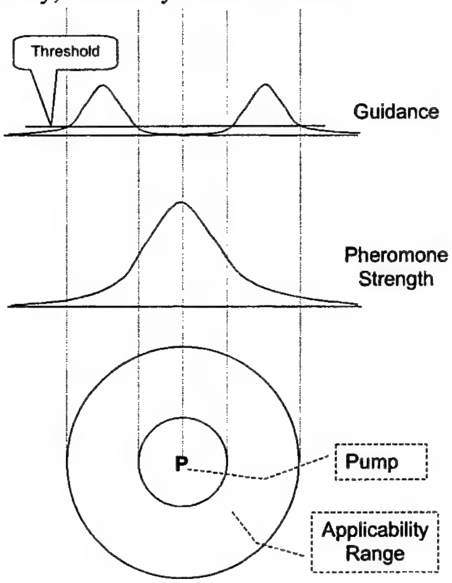


Figure 4. Hypothesized relation among pump location, pheromone field, and guidance

We hypothesized that a pheromone's applicability range for spatial guidance depends primarily on the propagation factor of the pheromone. In an area near the pump that creates the pheromone field, propagation loops back through the hexagonal grid to the source through many paths. Thus the reduction in strength of the propagated deposits might be too low to establish a sufficient gradient near a pump, though the pheromone field is high.

Far away from a pump the pheromone field is low and any propagated deposit that arrives there may already have lost too much of its strength to make a difference in the gradient. Then, the applicability range of the pheromone lies in the ring around the pump where the pheromone concentrations change the most with changing distance

4 Exploring the Hypothesis

To explore our hypothesis, we constructed a combinatorial analysis of the propagation of pheromones through a hexagonal grid and the guidance that results. This analysis challenges several key aspects of our initial hypothesis, but suggests a new explanation that is confirmed by experiment.

4.1 Combinatorial Analysis of Guidance

To understand what guidance the gradient in the local concentrations of a pheromone actually gives to a walker, we consider the spatial pattern of the concentration of a pheromone around one stationary pump. Assume, that the pump deposits one unit of the pheromone per unit time. This pheromone has an evaporation factor $E \in (0,1)$, a propagation factor $F \in (0,1)$, and a threshold $S \geq 0$. The remaining local concentration of the pheromone at a place after one evaporation step is E times its strength one unit time before. The propagation of a deposit event from a place to any of its direct neighbors in one propagation step is F times the strength of the deposit received divided by the number of direct neighbors, as long as it is larger than or equal to S.

4.1.1 Predicting Pheromone Concentrations

The spatial pattern of pheromone concentrations resulting from the pump's activities is symmetrically centered around the pump. Assume that the pump is located at a place p_0. On the basis of p_0 we structure the places of the hexagonal grid into disjoint sets P_d. Each set P_d comprises all places that are reached from p_0 in d steps on the shortest path. P_0 only contains the pump's place p_0, and P_1 is the set of all direct neighbors of p_0. In general, the set P_d comprises all direct neighbors of all places in P_{d-1} that are neither in P_{d-1} nor in P_{d-2}. The set P_d $(d>0)$ has $6d$ elements. Altogether, there are $6*(2d-1)$ links from elements in P_d to elements in P_{d-1}, $6*(2d)$ links to elements in P_d, and there are $6*(2d+1)$ links to places that are $d+1$ steps away from p_0.

A deposit of strength one by the pump at p_0 triggers a deposit of $F/6$ at every place in P_1. A deposit of strength $F/6$ at a place in P_1 triggers a deposit of $(F/6)^2$ at p_0, at two places in P_1, and at three places in P_2. In general, a deposit of strength s at a place in P_d triggers a deposit of $s*F/6$ at an average of $(2d-1)/d$ places in P_{d-1}, at two places in P_d, and at an average of $(2d+1)/d$ places in P_{d+1} $(d>1)$. Each propagation step is assumed to take one unit time. The sum of the propagated deposits to a place in P_d t time units after the deposit by the pump at p_0 is computed recursively as:

$$q(d,t) = \frac{F}{6}\begin{pmatrix} \frac{2d-1}{d-1}q(d-1,t-1)+ \\ 2q(d,t-1)+ \\ \frac{2d+1}{d+1}q(d+1,t-1) \end{pmatrix}$$

Since the pump repeats its deposit every unit time, a place in P_d receives a propagated input of $Q(d,t)=\sum_{j=0}^{t} q(d,j)$ at an arbitrary point in time t. Following the analysis of the pheromone infrastructure in [2], the pheromone concentration at a shortest distance of d steps from p_0 approaches the fixed point $B(d)=\lim_{t\to\infty}(Q(d,t))/(1-E)$. The graph in figure 5 shows the fixed point of the pheromone concentration on a logarithmic scale for varying distances and propagation parameters with an evaporation factor fixed to $E=1/10$. As a consequence of the cyclic nature of the hexagonal grid, we observe a rapid decline of pheromone concentrations as we move away from the pump.

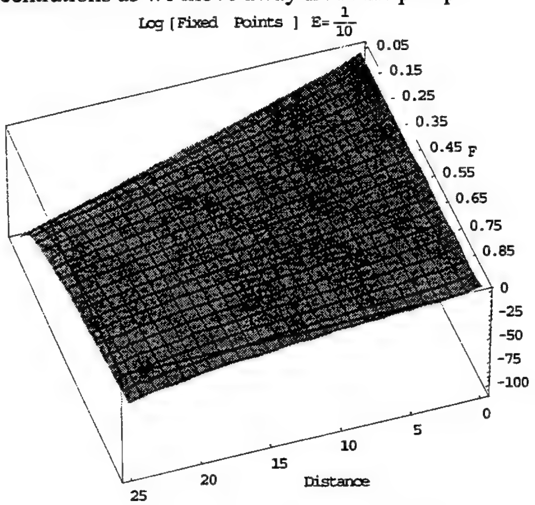


Figure 5. Fixed points of pheromone concentration (logarithmic scale)

4.1.2 Testing the Hypothesis

Applying this formalism, we now compute the local guidance available to a walker from a single pump as a function of both distance from the pump and propagation factor (Fig. 6).

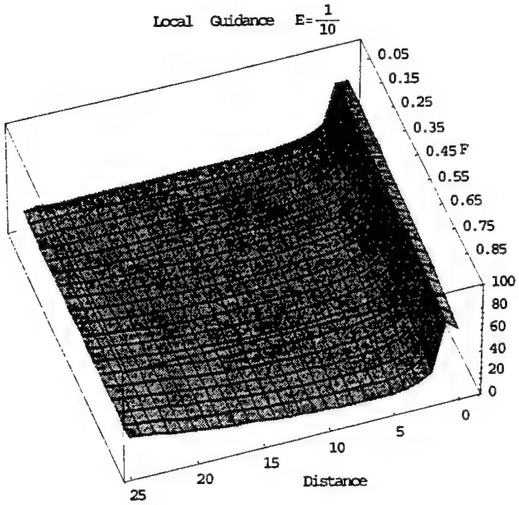


Figure 6. Local guidance without threshold

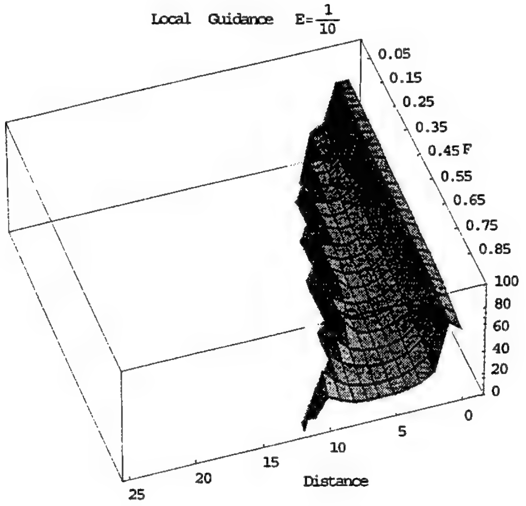


Figure 7. Local guidance with threshold

This picture is disconcerting to our hypothesis. A propagation-dependent applicability would appear as a ridge of high guidance, running from (low *d*, low *F*) to (high *d*, high *F*). Instead, guidance is greatest at the pump (*d=0*) or in the places adjacent to it (*d = 1*). Then it drops off rapidly. For low propagation the guidance is fairly constant with *d*. For high propagation, it drops somewhat lower than for low propagation, but only by a factor of two. Then it actually increases with increasing distance. This increase reflects the fact that the local guidance is an approximation of the second (spatial) derivative of the pheromone concentrations. As we see in Figure 5, the decline of the pheromone concentration *increases* with increasing distance.

What then accounts for the behavior we observed in our initial experiments? Those experiments explored the two pheromone parameters inspired by physical analogy, but did not examine the threshold, which we viewed simply as an implementation detail. However, when we apply the threshold to the local guidance, a cliff emerges running from low *d* and low *F* to high *d* and high *F* (Fig. 7). For large F we actually observe an increase in the guidance towards the cliff.

4.1.3 Local Guidance in a Field of Pumps

But is the observed increase in guidance towards the cliff at larger distances the predicted applicability range effect? Consider an arbitrary place p_i on the hexagonal grid. The local guidance available to a walker on p_i may be influenced by several pumps. A pump influences the guidance on p_i, if the propagation field of the regular deposits of the pump covers at least one of the places in $C(p_i)$, the options in a walker's relocation decision at p_i. The radius of the propagation field of a pump depends on the pheromone's propagation factor and the threshold. The last place to receive a propagated input is at a distance of $R_P=ln(S)/ln(F/6)$ steps away from the pump. Thus, if p_i is less than R_P+2 steps away from a pump, its local guidance depends at least on this pump's propagation field.

Assume there are currently n_i pumps that influence the local guidance at p_i. Then, the influence acts in two ways. On the one hand there is the distance of a pump from p_i, which, in relation to the other pumps' distances, determines the strength of the influence of this pump. On the other hand, the location of the place p_i in relation to the nearby pumps also influences the local guidance.

If n_i is zero, then there is no pheromone concentration at any of the places in $C(p_i)$ (assuming the concentrations have reached their respective fixed point). The relocation decision of a walker is a random selection of one of the seven places and the local guidance is zero.

If n_i is one, the local guidance at p_i depends on the distance from a single pump. As we have seen in Figure 7, the guidance has local maxima very close to the pump and at the outer limit of the propagation field.

Finally, if n_i is larger than one, several pumps influence the local guidance at p_i. We have been able to identify some scenarios of low or high local guidance, but a complete numerical prediction remains yet to be found.

If the place p_i is significantly closer to one of the n_i pumps than to all the others, the local guidance at p_i is dominated by this nearest pump. In this case the local guidance is predicted as in the single pump discussion. Our observations show that a difference of two or more steps in the distances of the n_i pumps from p_i is already sufficient to return to the single pump case.

If all n_i pumps are about the same distance away from p_i, then the local guidance depends on the location of p_i in relation to these pumps. If most pumps are in the same direction from p_i, then their effect is again similar to a single pump at the average distance of these pumps. On the other hand, if the place p_i is surrounded by the pumps, their guidance effect is diminished. An extreme scenario, exemplifying this diminishing effect is a pump at each direct neighbor of p_i. In this case, the pheromone concentration at six places in $C(p_i)$ is X and at one place it is Y, with $X >> Y$. The local guidance is reduced to approximately $1/6-1/7=1/42$.

4.1.4 A Confirming Experiment

The previous discussion predicted a primary dependency of the local guidance in a field of pumps on the pheromone's propagation factor F and on its threshold S. Secondarily, there is also a dependence on the location of the respective place in relation to the pumps whose propagation field cover the place or one of its direct neighbors. The second influence is only secondary, because it is the radius of the propagation field, determined by F and S, that allows multiple pumps to influence a place in the first place.

This observation leads to the following new hypothesis on local guidance and the applicability of a pheromone for spatial coordination:

A pheromone is suitable for spatial coordination of walkers (high local guidance) in the close neighborhood of pumps (about 5 steps), if it has a small propagation radius. It serves walkers at a medium distance from the pumps (about 15 steps), if it has a large propagation radius. Walkers at larger distances away from any pump cannot be guided by propagated pheromones, because the explosion in the required propagation out to such a distance.

A small propagation radius requires a relatively large threshold S or a small propagation factor F, whereas a large propagation radius is achieved with small S or large F. The following figure (Fig. 8) shows the local guidance in the case of our field of fifty pumps for combinations of propagation factors $F=1/10$ and $F=9/10$, and thresholds $S=10^{-2}$ and $S=10^{-6}$. The plots show the best guidance in the regions near the pumps for the configuration $F=1/10$ and $S=10^{-2}$. The best guidance in the medium-distance areas is available for $F=9/10$ and $S=10^{-6}$.

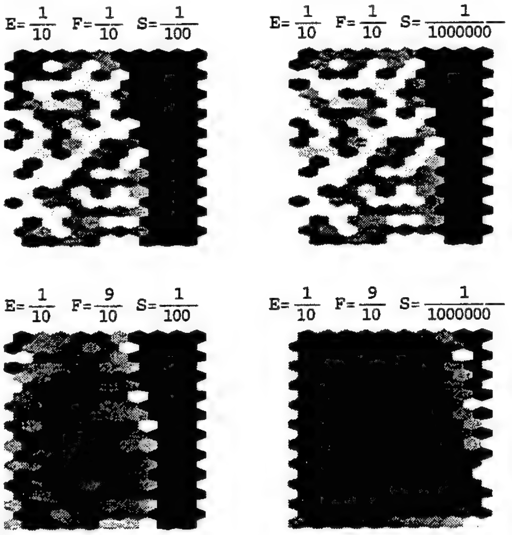


Figure 8. Local guidance for varying pheromone configurations

The underlying assumption in the prediction of the local guidance for a specific pheromone configuration is that there is a good guidance at places that are just at the outer limit of the propagation range of a small number of pumps. To verify this assumption, we plot for each place on the grid the number of pumps that are exactly at a distance of R_P from the respective place (coverage). Such a plot should indicate areas of potentially high guidance for a given pump distribution and a specific pheromone configuration (F and S). Figure 9 shows the plots for four pheromone configurations.

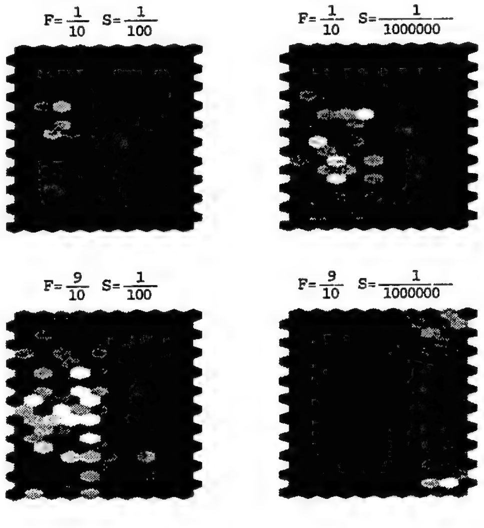


Figure 9. Local coverage for varying pheromone configurations

Comparing the local guidance plot of (Fig. 8) with the coverage plots (Fig. 9), we see a link between high guidance and medium coverage of a place. That high coverage does not automatically mean high guidance may seem counter-intuitive at first. But then, with increasing coverage there is also an increasing risk that the pumps are located in different directions from the place. As our previous discussion indicates, pumps at different directions of a place actually decrease the local guidance. The risk of having pumps located at about the same distance but in different directions increases with the number of pumps that significantly influence the local guidance of a place.

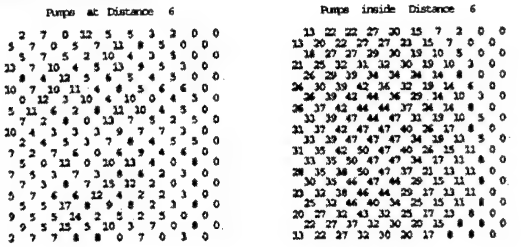


Figure 10. Influence of pumps at R_P=6

Figure 10 illustrates this effect for a propagation radius of six steps. The left plot shows for each place the number of pumps that are exactly six steps away, while the right plot shows the number of pumps that are maximally six steps away. From the right plot we can derive an indication, where places with good guidance for a pheromone propagation radius of six are. Good guidance is expected to be at places with a small, but non-zero number of influencing pumps. The number may be larger the farther a place is outside a cluster of pumps.

5 Configuring Pheromones

The local guidance available to a walker in its relocation decision depends on its current distance to the pumps, the propagation radius of a pheromone, and the spatial distribution of the pumps. Even with stationary pumps, as we have considered them in this paper, it is obvious that one pheromone configuration cannot provide good guidance at all places. We need more variety in our pheromone vocabulary to improve the performance of the walkers.

Our enhanced vocabulary comprises pheromones with different propagation radii. Thus, we have pheromones that provide guidance near the pumps, while other pheromones guide walkers at medium distances. Walkers that are a long distance away from the pumps will have to rely on random walk until we design a different guidance mechanism for them.

The behavior of the pumps and the walkers is adapted to the enhanced vocabulary. Pumps regularly deposit a collection of pheromones, one for each specified pheromone configuration. All deposits have the same fixed strength.

A walker is able to perceive pheromones of different configurations separately. Thus, it is able to decide what pheromone to use for its probabilistic selection in its current relocation step. The walker always follows the gradient of the pheromone that currently has the highest local guidance at the place of the walker. Thus, it will automatically employ the pheromone most appropriate to its current location in relation to the pumps. In addition, the configuration of the selected pheromone allows the walker to estimate its current distance to the pumps.

The choice of the most appropriate pheromone vocabulary is guided by the availability of communication and processing capacity in the execution system, as well as by the typical spatial distribution of the pump population and its relation to the walkers.

The most straightforward choice would be a pheromone configuration for each propagation radius between one and approximately fifteen steps. Assuming all pheromones share the same propagation factor *F*, the required threshold S_r for a given propagation radius *r* is computed as $S_r=(F/6)^r$. However, practically speaking, pheromones with larger propagation radii often convey sufficient guidance at more than one distance away from the pump. Thus, the vocabulary may be reduced to save communication and processing resources.

Figure 11 shows the combined guidance for a vocabulary of six pheromone configurations in the case of our pump population, including pheromones with propagation radii between one and six. As specified before, a walker always picks the pheromone with the maximum local guidance. As the plot shows, now there is good guidance at places near the pumps as well as at medium distance places.

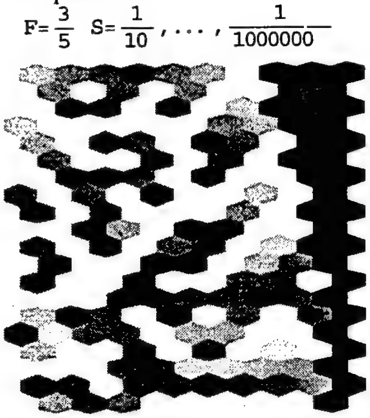


Figure 11. Combined local guidance for propagation radii R_P=1,...,6

An adaptive approach automatically strikes the balance between complete coverage of the vocabulary space and optimization of the execution performance. Initially, the pumps deposit pheromones configured for all possible propagation radii. In addition to moving towards the pumps, each walker keeps a profile of the usage of the different pheromone configurations, and reports this profile regularly to the pumps it meets. Pheromone configurations that are seldom used either cover areas where walkers never reach, or convey low guidance. Such configurations have a higher chance of being dropped from the active vocabulary of the pumps. To accommodate possible changes in the system's dynamics, configurations that have been dropped are introduced back into the vocabulary at random intervals.

6 Experimental Performance Evaluation

Finally, we evaluate the expected improvement in the performance of the walker population. In our experiment we compare the three relocation strategies we presented in this paper. The baseline performance is measured in the random selection of the next location. Then, there is the probabilistic, pheromone-biased selection, always following the guidance of the same pheromone. Finally, we specify six separate pheromone configurations, resulting in propagation radii between one and six. A walker first selects the pheromone with the highest local guidance and then its next location following this pheromone's gradient.

Since different individual walkers move independently, it is sufficient to observe only one of them as it walks over the grid. The walker is placed randomly on the 10x10 hexagonal grid and is permitted to relocate one hundred times. We repeat this experiment one hundred times, each time with a different random seed, to capture statistically significant data.

The pump population of fifty individuals is placed on the grid as in our previous discussions. Each pump deposits a pheromone from each configuration with one unit strength each unit time.

We measure the performance of the walker as the average number of pumps with which the walker shares the same place in each cycle. We call this metric the walker's co-location number.

Theoretically, the best possible co-location number in the chosen setup of the pump population is five pumps, since there are two places with five pumps. But this ignores the random initial placement of the walker, since a walker has to spend some time before it may get to a place with five pumps. Then there are eight places with three and eight places with two pumps.

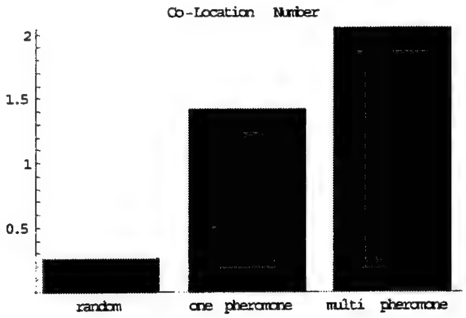


Figure 12. Effect of relocation strategies

In figure 12 we plot the co-location number observed for the three different relocation strategies. In the chosen configuration a random walker shares a place with an average of 0.26 pumps. A walker that always follows the pheromone with a propagation radius of three, shares its place with an average of 1.41 pumps. This is an improvement of a factor of 5.4 compared to the random baseline.

A walker that takes all six pheromones into account, achieves a co-location number of 2.05 pumps. Thereby it performs 7.9 times better than random. The improvement against the one-pheromone relocation strategy is still significant with a factor of 1.5. Thus, the experiment yields the predicted improvement in the walker's performance.

7 Conclusion

Pheromone systems are a simple, robust mechanism for generating spatial guidance and coordination among mobile agents. Their robustness is due largely to their nonsymbolic, quantitative nature. Their behavior, like that of other such mechanisms, is sensitive to various tuning parameters.

We have found the concept of "guidance" to be a simple but powerful help in tuning these parameters. This metric, a form of gradient of the field, estimates how much direction the pheromone field gives an agent in deciding on its next step. Common analyses of pheromone fields focus on the absolute strength of the field, but for some purposes the guidance, which can be high where the field is weak or low where the field is strong, is more strongly correlated with agent performance.

Focusing on the guidance metric shows that a single pheromone is a compromise between short-range and long-range direction of a mobile agent. In the context of air operations, for example, a pheromone that can best lead aircraft to the general vicinity of targets will be ill-suited for guiding them to individual targets. Our experiments and analysis show that the best guidance over a range of distances is achieved by using multiple pheromones with varying propagation rates and thresholds. Agents change their focus from one pheromone to another based on which one offers the highest guidance at their current location.

Multiple pheromone methods significantly improve the performance of agents over a range of distances, without compromising the simplicity and locality of interaction that recommend the pheromone approach to spatial guidance and multi-agent coordination.

8 References

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz. Swarm Intelligence: From Natural to Artificial Systems. Santa Fe Institute Studies in the Sciences of Complexity. Oxford University Press, 1999.

[2] S. Brueckner. Return from the Ant – Synthetic Ecosystems for Manufacturing Control. PhD thesis, Humboldt University Berlin, Germany, 2000. (forthcoming).

[3] P.-P. Grasse. La reconstruction du nid et les coordinations inter-individuelles chez bellicositermes natalensis et cubitermes sp. la theorie de la stigmergy: Essai d'interpretation du comportement des termites constructeurs. Insectes Sociaux, 6:41-80, 1959.

[4] H. V. D. Parunak. 'Go to the Ant': Engineering Principles from Natural Agent Systems. Annals of Operations Research, 1997.

[5] H. V. D. Parunak. Adaptive control of Distributed Agents through Pheromone Techniques and Interactive Visualization. 2000. Web Page, www.erim.org/cec/projects/adaptiv/.

A Design Space for Enterprises

Sarosh Talukdar, Eduardo Camponogara and Haoyu Zhou
Carnegie Mellon University, ECE Department, Pittsburgh, PA 15213
talukdar@cmu.edu, camponog@cs.cmu.edu, haoyu.zhou@cs.cmu.edu

Abstract

The performance of an enterprise depends on the capabilities of the agents that control it and on how these agents are organized. The division and coordination of labor are two of the principal functions of an organization. This paper assembles a five-dimensional space from which organization designers can select coordination policies for a wide class of distributed enterprises. The assembly is based on the extension of model-predictive control to distributed agents, and on the conjecture that distributed enterprises are reducible (meaning that their control problems can be broken into much smaller sub-problems.) This conjecture has not been verified, nor has the space of coordination policies been explored. But samples from the space show promise. They allow agents to work asynchronously--in parallel, each at its own speed--and generate solutions that dominate Nash equilibria.

1. Introduction

1.1 Terminology

The enterprises considered here, are those, such as the electric grid, telecommunications networks and traffic networks, that contain, and are controlled by, distributed organizations; where: An organization is a network of agents and communication links, and a set of policies for operating this network. An organization is distributed if its agents are local, that is, if each agent can sense only a few of the enterprise's state variables and has authority over only a few of the enterprise's control variables. An agent is any element of a continuum of decision-makers stretching from the simplest relay (a device that decides when a single measured variable crosses a threshold), to organizations of intelligent robots and humans. (In other words, greater agents are organizations of lesser agents.) And operating policies are strategies by which organizations perform complex control (large dynamic optimization) tasks through the division of these tasks among their agents and the coordination of the agents' work. In other words, a distributed organization is a quadruple:

$\Phi = \{A, L, d, c\}$

where A is a set of M local agents represented by the nodes of a graph, L is a set of communication links represented by the arcs of the graph, d is a policy for the division of the organization's task among the agents, and c is a policy for the coordination of the work of the agents.

1.2 Organization Design Spaces

The performance of an organization depends not only on its agents but also on its other components, L, d and c. In other words, the same set of agents may be expected to perform quite differently when their organizations are changed, say from free markets to rigid hierarachies.

Organizations for natural agents, such as humans, insects and cells, have been under study for many years. A good deal is known about their design. In contrast, relatively little is known about the organization-design-spaces for software agents. (A design space is the set of structural alternatives that the designer considers in the search for a good design.)

Organizations for software agents differ from those for natural agents in their coordination policies. These policies depend on the agents' social characteristics (such as openness to suggestion, trustworthiness, the propensity for reciprocal altruism, and the frequency of interaction). The social characteristics of natural agents are fixed, or at least, difficult to change. But the social characteristics of software agents are programmable. Therefore, the organization-design-spaces for software agents may be expected to differ from, and probably be larger than, the spaces for natural agents.

1.3 The Problem

The remainder of this paper is devoted to the problem: given P, A, L and d, that is, given the task to be performed by, and the first three components of, a software organization, assemble a design space (a set of alternatives) for the fourth component, c, the coordination policy. More specifically, Section-2 describes an extension of model predictive control by which a large dynamic control task can be decomposed into smaller, static optimization tasks, one for each agent. Section-3 explains the goals of policies for coordinating the work of the agents on these smaller tasks. Section-4 lists the dimensions of a space for coordination policies. Section-5 reviews some open issues.

2. Distributed Model Predictive Control

Any dynamic optimization problem, DOP, can be approximated by a series of static optimization problems, $\{SOP_j\}$, through the invocation of a discrete variable technique that uses state-prediction and overlapping time intervals. This technique is known by various names including "model predictive control" [1] and "rolling horizon planning" [2]. Each interval extends from the current time to a "horizon in the future". To be more specific, let:

- $T_j = [tj_1, tj_2, ---, tj_K]$ be discrete points in time covering the interval that extends from "now" (time $= tj_1$) to the "j-th horizon" (time $= tj_K$),
- x be the vector of the state and control (decision) variables of DOP, and
- X be the vector of the discrete values of x over Tj, that is: $X = [x(tj_1), x(tj_2),---, x(tj_K)]$.

Then, the derivatives and integrals in DOP can be replaced by discrete approximations in terms of X, to yield a static approximation, SOP_j.

The disadvantages of using a single approximation, SOP_j, to calculate the real-time controls of a large enterprise are twofold. First the calculations are "open loop" (they use model-based-predictions of future states, and these predictions become less accurate the further one goes into the future). Second, SOP_j is often unmanageably large (it has K-times as many state and control variables as DOP).

The usual way of countering the first disadvantage is to introduce feedback by periodically "rolling the horizon forwards," that is, by recalculating the static approximation for a series of overlapping time intervals. (Only the very first part of the control plan obtained by solving SOP_j is implemented; after a short time has passed, a new approximation, SOP_{j+1}, is formulated, initialized with current state measurements and solved to obtain a new control plan; the process is repeated through SOP_{j+2}, SOP_{j+3}, ---.)

In what follows, we propose dividing the agents into overlapping sub-sets, called neighborhoods, to eliminate the second disadvantage.

2.1 Reducibility, Quality and Tractability

The general form of SOP_j is a multi-objective, mixed integer, nonlinear optimization problem [3].

$$
\begin{array}{ll}
SOP_j: & \underset{X}{\text{Minimize}}\ F(X) \\
& \text{subject to: } G(X) \le 0 \\
& \qquad\qquad\quad H(X) = 0 \\
& \qquad\qquad\quad X \text{ in } S
\end{array}
$$

where F, G and H are function vectors, and S represents the integer constraints, if any, on X.

Consider the m-th agent in a network of M agents. Define the neighborhood of this agent to be the set of agents that are adjacent to it. Then, SOP_j can be decomposed into a collection of single objective sub-problems, one for each agent, such that the m-th sub-problem has the form:

$$
\begin{array}{ll}
p_{jm}: & \underset{X_m}{\text{Minimize}}\ f_m(X_m, Y_m, Z_m) \\
& \\
& \text{subject to: } g_m(X_m, Y_m, Z_m) \le 0 \\
& \qquad\qquad\quad h_m(X_m, Y_m, Z_m) = 0 \\
& \qquad\qquad\quad X_m \text{ in } S_m
\end{array}
$$

where X_m is a vector of agent-m's local variables over the time interval T_j; Y_m is a vector of agent-m's neighbors' local variables over T_j; Z_m is vector of all the other variables in X; f_m is a scalar objective, g_m and h_m are function vectors, and S_m represents the integer constraints, if any, on X_m. In other words, $X = \text{Union}(X_m, Y_m, Z_m)$, $F = \text{Union}(f_1,---, f_M)$, $G = \text{Union}(g_1, ---, g_M)$, $H = \text{Union}(h_1, ---, h_M)$, and $S = S_1XS_2X---XS_M$.

For large enterprises, problem SOP_j tends to be intractable--it has too many variables and constraints. But problem p_{jm} can be made tractably small. (We believe that many, if not all, distributed enterprises are reducible in the sense that their variables and constraints can be ordered so p_{jm} can be made much smaller than SOP_j and insensitive to Z_m, for all j and m).

Even though $\{p_{jm}\}$ is an exact decomposition of SOP_j, the simultaneous solutions of $\{p_{jm}\}$ are not necessarily the best solutions of SOP_j. To explain, we need three concepts from game theory [4]. Specifically, let:

- R_m, the reaction set of agent-m, be the decisions that agent-m should make when it knows what all the other agents are going to do. More formally:
  $R_m = \{X_m(Y_m,Z_m)$ such that X_m is an optimum solution of $p_{jm}\}$

- N, the set of Nash equilibria, be the intersection of the reaction sets of all the agents
- P, the Pareto set of SOP_j, be the set of feasible solutions of SOP_j that are not dominated by any other feasible solutions. A solution is feasible if it meets all the constraints. A solution, Xa, dominates another solution, Xb, if $f_m(Xa) \leq f_m(Xb)$ for all m, and if $f_m(Xa) < f_m(Xb)$ for at least one m.

Three observations are:

- The elements of the Pareto set represent the best possible tradeoffs among the multiple objectives of SOPj--better tradeoffs than are provided by Nash equilibria, as illustrated in Figs. 1 and 2.
- Constraints can change the solution sets in profound ways (compare Figs. 1 and 2). While it is always possible to invoke techniques, such as Lagrange multipliers, penalty functions and barrier functions, to convert a constrained problem into an unconstrained one, we believe that such conversions should be used, if at all, with caution. There are both conceptual and computational advantages to preserving the separate identities of constraints, not the least of which is the option of specialized, adaptive handling of each constraint during the solution process.
- SOP_j and p_{jm} represent two extremes of a continuum of problems. SOP_j tends to be intractable but its (Pareto) solutions are the best that can be obtained. p_{jm} is smaller and much more tractable; the collection of its solutions for different values of Y_m and Z_m constitute its reaction set; and the intersection of all these reaction sets identifies the Nash equilibria of SOP_j. But the calculation of a reaction set requires the repeated solution of p_{jm}, which can be tedious.

3. Coordination Goals

Camponogara [5] has devised a coordination policy that provides a short-cut for the calculation of Nash equilibria. If:

- SOP_j is feasible
- SOP_j is convex
- SOP_j is reducible (Z_m is empty for all m)
- p_{jm} is solved by an iterative interior point algorithm that uses the latest available estimates of Y_m
- the agents in each neighborhood perform their iterations sequentially, passing the result of each iteration, as soon as it is obtained, to their neighbors,

then the iterations will converge to a Nash equilibrium of SOP_j. On the plus side, this short cut makes it unnecessary to compute the reaction sets exactly ("first-iterate" approximations to small portions of the reaction sets are sufficient). On the minus side, the short-cut requires convexity (most real problems are non-convex), complete reducability, restrictions on parallel work (only agents not in the same neighborhood are allowed to work in parallel), and the short-cut produces Nash equilibria, not Pareto solutions.

4. A Design Space For Coordination Polices

Empirical investigations indicate that the conditions on convexity and reducibility are unnecessary, and can be relaxed in small but representative enterprises. Assuming these relaxations will scale to larger enterprises, the remaining goals for coordination policies are:

- allow for asynchronous work, that is, make it possible for all the agents to work in parallel, each at its own speed,
- obtain better solutions than Nash equilibria.

This section translates these goals into requirements, proposes heuristics for meeting these requirements, and thereby, assembles a space of coordination policies.

4.1 Asynchronous Work

The two main requirements for asynchronous work are:

- Each agent needs to know what the other agents are going to do. One way to meet this requirement is to include automatic learning schemes (such as neural nets and decision trees), so each agent can transform historical data into predictions of future actions. The learning can be used in two ways. The first, is for each agent to predict what its neighbors will do , that is, the value of Y_m. The second, is for each agent to estimate what it will do, that is, the value of X_m, and communicate this estimate to its neighbors before it has finalized the value of X_m.
- Each agent needs to know what the other agents might want to do, so it can allow for these wants, if it chooses to be unselfish. Shared resources (such as an electric transmission line of limited capacity over which the power produced by several competing generators must flow) are of particular concern. Such resources need to be shared in ways that, in hindsight, will seem fair. But there is a risk that the faster agents will grab more of the resources than they rightly deserve. One way to reduce this risk is to use "resource margins," that is, to replace the constraint: $g_m(X_m, Y_m, Z_m) \leq 0$, by the tighter constraint: $g_m(X_m, Y_m, Z_m) \leq -r_m^2$.

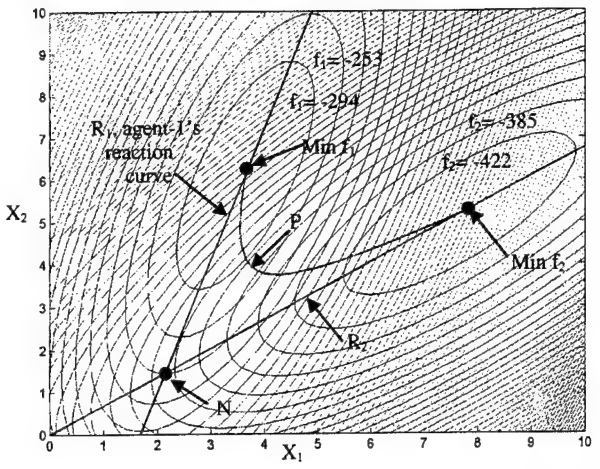


Figure 1. The objective landscape for a two-agent organization with:

$$p_{j1}:\ \underset{x_1}{\text{Min}}\ f_1(x_1,x_2) = 89.51x_1^2 - 57.74x_1x_2 + 20.48x_2^2 - 150x_1 - 50x_2$$

$$p_{j2}:\ \underset{x_2}{\text{Min}}\ f_2(x_1,x_2) = 38.97x_1^2 - 68.95x_1x_2 + 51.02x_2^2 - 120x_1$$

where $f_1(x_1,x_2) = 89.51x_1^2 - 57.74x_1x_2 + 20.48x_2^2 - 150x_1 - 50x_2$, and $f_2(x_1,x_2) = 38.97x_1^2 - 68.95x_1x_2 + 51.02x_2 - 120x_1$. The ellipses are level sets of f_1 and f_2.

4.2 Improving on Nash Equilibria

In order to improve on Nash equilibria, it would seem necessary for agent-m to take a broader, and perhaps, more altruistic view than is provided by problem p_{jm} and the solutions that constitute its reaction set. There are at least two ways of taking such a view. The first, is by augmenting p_{jm} to include some of the criteria (objectives and constraints) of other agents. The second, is for agent-m to cede control of some of its variables to other agents.

4.3 Four Dimensions

To summarize, four dimensions of a design space for coordination policies are:

- Learning (processes by which agent-m can obtain early estimates of X_m--to help its neighbors in their decision-making--and to predict Y_m, to help itself in its own decision-making).
- Fairness (the choice of r_m, a means for tightening the constraints on the resources that agent-m must share, to keep this agent from over using these resources).
- Altruism (the degree to which p_{jm} is augmented to include the criteria--objectives and constraints--of other agents).

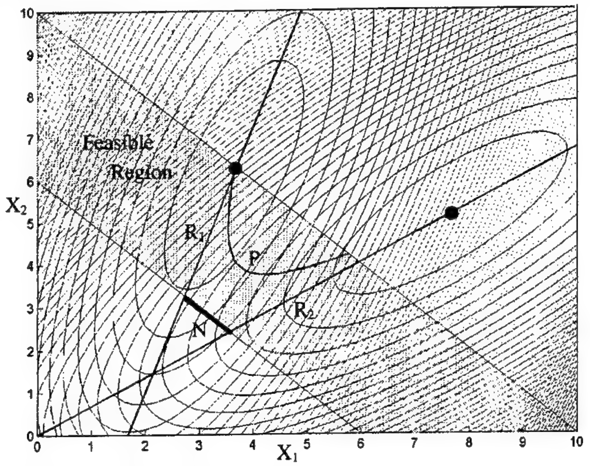


Figure 2. The objective landscape for a two-agent organization with:

$$p_{j1}:\ \underset{x_1}{\text{Min}}\ f_1(x_1,x_2)$$
$$\text{s.t.}\quad x_1+x_2 \geq 6$$
$$x_1+x_2 \leq 10$$

$$p_{j2}:\ \underset{x_2}{\text{Min}}\ f_2(x_1,x_2)$$
$$\text{s.t.}\quad x_1+x_2 \geq 6$$
$$x_1+x_2 \leq 10$$

where f_1 and f_2 are as in Fig. 1.The ellipses are level sets of f_1 and f_2.

- Deference (the degree to which agent-m cedes control of its decision variables to other agents).

These are not the only dimensions of coordination. Nor have we even explored the space they span. But results from some arbitrarily selected members of this space show promise, such as the improvements over Nash solutions illustrated in Fig. 3.

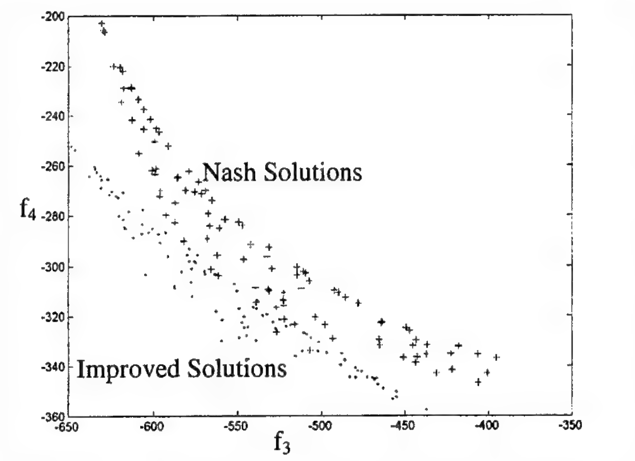


Figure 3. A two-dimensional projection showing the improvements in solutions obtained by a four-agent organization using a coordination policy with some "fairness,""altruism" and "deference," but no "learning."

5. Concluding Remarks

The top-down approach to organization design taken in the paper has produced a space of coordination policies for distributed enterprises, such as electric grids and telecommunication networks. But this space remains to be examined, as does the conjecture on which it is based. As such, the work reported here can be viewed as a specification of research questions, including:

- Are distributed enterprises reducible, as conjectured in the paper? If so, to what extent?
- What paths through the coordination-policy-space produce the best tradeoffs between solution-quality and computational effort?
- How should the other components of software organizations (A, L and d), be designed?

Acknowledgement

This research was supported in part by the Pennsylvania Infrastructure Technology Alliance, EPRI/ARO under contract number EPRI-W08333-05, and the Brazilian National Council for Research and Development (CNPq) under grant number 20.0047/96-5.

References

[1] E. Mosca, *Optimal, Predictive, and Adaptive Control*, Prentice-Hall, 1995.

[2] T.E. Morton and D.W. Pentico, *Heuristic Scheduling Systems*, John Wiley and Sons, New York, 1993, pp. 221-223.

[3] K.M. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer Academic Publishers, 1999.

[4] T. Basar and G.J. Olsder, *Dynamic noncooperative game theory*, Academic Press, 1982.

[5] E. Camponogara, "Controlling Networks with Collaborative Nets," Ph.D. Dissertation in progress, ECE Department, Carnegie Mellon University, 2000.

Invited Talk 2

Deployability Analysis at the Military Traffic Management Command Transportation Engineering Agency

Deployability Analysis at the Military Traffic Management Command Transportation Engineering Agency

Joe W. Knickmeyer
Chief, Systems Integration Division
Military Traffic Management Command
Transportation Engineering Agency

ABSTRACT: The Military Traffic Management Command (MTMC) is one of three Transportation Component Commands under the U.S. Transportation Command (US TRANSCOM). In turn, one of MTMC's component commands is the MTMC Transportation Engineering Agency (MTMCTEA). Located in Newport News, Virginia, the Agency is responsible for assuring that equipment designed for military use is transportable by available means of movement to any location required, that the worldwide transportation infrastruture is capable of accommodating our forces, and that the transport vehicles and methods available permit projecting US military might wherever it is needed, on time. Current goals for the rapidity of deployment are ambitious and difficult. Dramatic improvements in all aspects of the Defense Transportation System will be required to meet these goals. The presentation describes the analytical tools the Agency uses to support assessment of the capability of the transportation system, and to support the development of war plans by the geographical Commanders in Chief (CINCs). These tools are a subset of the total array available to Defense analysts, but they are typical in style and intent, if unique in their level of detail. Most of these tools are simulations or schedulers of movement and asset allocation; some have a network analysis component, but none currently in use explicitly model the information flow or capture the impact of failures in such flows. We touch on the formal command and control systems in place, under development, or proposed to monitor and control deployment operations, and provide a short overview of the Advanced Logistics Program (ALP) and its potential contribution to deployment operation control.

Introduction

The United States finds itself today in a world fundamentally different from that in which our former strategies of forward deployed forces and massive nuclear capability proved so successful. The arrival of asymmetrical warfare, rising political and economic chaos in the wake of the collapse of the Former Soviet Union, the breakdown and failure of national government systems, and increasing religious and ethnic hostilities have created an unstable and dangerous world. The armed forces of the United States have found their missions multiplied, the range of necessary responses expanded, and the need for rapid and agile deployment more critical than ever before. At the same time, the force is compelled to become more efficient and lean both from economic necessity and the need to project any significant power from CONUS bases far from the point of application.

The Military Traffic Management Command Transportation Engineering Agency (MTMCTEA) is responsible for ensuring the rapid deployability of US Forces world-wide, by detailed technical evaluation and enhancement of all aspects of the Defense Transportation System (DTS). The DTS is unusual, in that the Government and the Department of Defense directly control only a relatively small portion of the assets and infrastructure needed for power projection. MTMCTEA concerns span the range from design of equipment and weapon systems for swift and efficient transportability, through the physical characteristics of installations, air and sea ports of embarkation and debarkation, and transportation networks in CONUS and abroad, to the techniques and procedures used to coordinate and control Defense movements. This broad mission is carried out by a small staff of civil and mechanical engineers, operations research and systems analysts, and computer engineers.

Organizationally, the Agency is a major subordinate command of the Military Traffic Management Command (MTMC), the Army component of the U. S. Transportation Command (USTRANSCOM). USTRANSCOM's other components are the US Military Sealift Command, responsible for ocean movement, and the Air Mobility Command, responsible for strategic air transportation operations worldwide. MTMC is responsible for surface movement of military goods and people within the Continental United States (CONUS), and for the operation of ocean terminals throughput the world.

Functionally, the Agency is a transportation engineering analysis arm of the US military structure. MTMCTEA's customer base ranges from the soldier in the field to the highest level planners in the Department of Defense. The following abbreviated list will give some sense of the scope of the organization's workload [MTMCTEA 00]:

- We develop detailed guidance, to include written reference manuals designed to be carried in field uniform pockets, to soldiers performing shiploading operations or tying down equipment on railcars.

- We determine the capability of the CONUS transportation infrastructure (highway, rail, waterway, origin installation or depot, and ocean terminal) to support military force projection operations.

- We support force modernization through analysis of the impact on deployability of conceptual and new weapon systems as well as changes in the force structure itself (e.g., Army XXI and Army After Next).

- We support the theater Commanders in Chief (CINCs) by determining the capability of host nation port, road and rail facilities to support US military movements both in conflict and in operations other than war.

- We provide direct support and modeling and simulation analysis to strategic exercises such as Atlantic Command's Unified Endeavor.

- We support programmatic analysis at the Joint Staff and Office of the Secretary of Defense level through analysis of strategic and tactical lift assets and enabling units. Examples of this work are the recurring Total Army Analyses and the Mobility Requirements Study.

To accomplish these missions, MTMCTEA necessarily maintains a formidable analytical capability in its M&S backbone. Support for transportability engineering includes finite element analysis, dynamical systems analysis models, and neural network and other artificial intelligence technologies targeted toward understanding how Defense equipment items respond to the transportation environment. The bulk of these engineering tools are commercial or government off-the-shelf software, as are packages employed by MTMCTEA traffic engineers to manage the safe and efficient flow of vehicles on DoD installations. The larger, integrated picture of force movement from origin to tactical assembly area in the objective theater requires a different set of tools.

The Deployability Problem

The sketch below (Figure 1) shows a schematic representation of the movement flow and important nodal points in military deployment. We'll use this simple display to highlight the applications both of modeling and simulation tools, and the command and control systems employed to manage the operational flow of a deployment. The operation begins, for us, at the origin installation. Units outload their equipment and personnel here by convoy, commercial motor and highway, and rail. At MTMCTEA, we do not model the movement to the origin installation of reserve units falling in on their mobilization stations—this is a Forces Command responsibility. We do hope at some future date to be able to share information on those flows in order to evaluate a total demand on the transportation system, but the impact on strategic operations is relatively minor. Also, the focus at MTMCTEA is on the flow of materiel, as opposed to personnel, since it is equipment movement that provides engineering challenges.

From the origin installation, the unit equipment flows to air and sea ports of embarkation, where the cargo will be loaded aboard strategic transportation vehicles: the commercial and Military Airlift Command airlifters, and the ocean vessels managed by the Military Sealift Command. MTMCTEA analyzes in detail the flow through the rail, highway and waterway networks to both air and sea terminals. We also analyze in detail the movement of cargo through water terminals; air port operations are not part of our mission, but are treated by MAC.

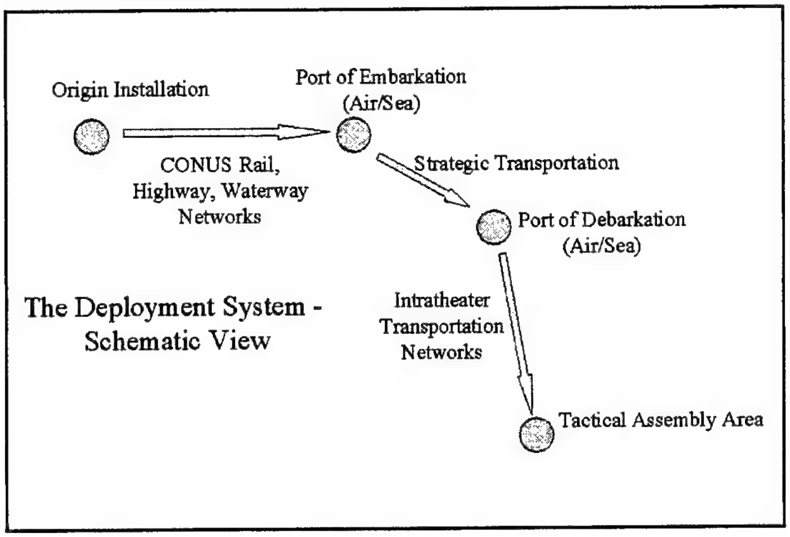

Origin Installation
Port of Embarkation (Air/Sea)
CONUS Rail, Highway, Waterway Networks
Strategic Transportation
Port of Debarkation (Air/Sea)
The Deployment System - Schematic View
Intratheater Transportation Networks
Tactical Assembly Area

Figure 1 – Schematic View of Deployment System.

The strategic transport vehicles move their cargo to air and sea ports of debarkation in or near the destination theater. Again, MTMCTEA focuses its analytical attention on the water terminal operations, for which our parent command has responsibility.

Intratheater movement is the responsibility of the theater CINC, who arranges that movement with his own transportation assets, supplemented by the assets of the host nation where such cooperation exists. The total theater transportation capability is thus dependent on the movement into the theater of military transport units and equipment, and varies over time. MTMCTEA assists the CINC with the analysis of the movement requirements imposed by his operation plan, and determines the capability of the theater to move the units to their tactical assembly areas, possibly through one or more intermediate staging bases. We do not concern ourselves with tactical movements involved in the combat operations.

The amount of time required to close forces sufficient to bring about decisive domination of the enemy is key. Buildup for ground operations in the Gulf War required many months. The current objective defined by the Army Chief of Staff, GEN Shinseki, is to deploy an initial brigade in 96 hours, a fully operational division in 120 hours, and a five-division force capable of sustained combat operations ready for employment in 30 days [Shinseki 99]. This ambitious goal is a major challenge to everyone working in the force deployment community, from those weapons systems acquisition to researchers in innovative transportation vehicles such as fast sealift ships.

Force Deployment Modeling

The centerpiece of the MTMCTEA deployability analysis suite is the Force Projection Model (FPM), a blanket title for a series of link and node models of the DTS. The nodes of the modeled system are transshipment points: the origin installation, air and sea ports of embarkation and debarkation, and the theater destination. The links are the rail, highway, air and sea routes that connect these nodes. Today, these models cooperate through the chain mechanism of an Expanded Time Phased Force Deployment File (TPFDD). The identity of individual equipment items is injected and maintained throughout the analysis, and movement status data recorded on the TPFDD, permitting highly detailed tracking of the location and state of individual items as the simulation progresses. Each model in the chain reacts to the throughput of the system immediately preceding it. While this level of integration is significant, reflecting a new precision in unit equipment item deployment modeling and planning, our eventual goal is to have all the FPM models fully High-Level Architecture compliant, running in a network web that bears considerable resemblance to the actual DTS network. Figure 2 below shows the models of the FPM suite, together with other commonly used tools that expand the analysis to all aspects of the DTS.

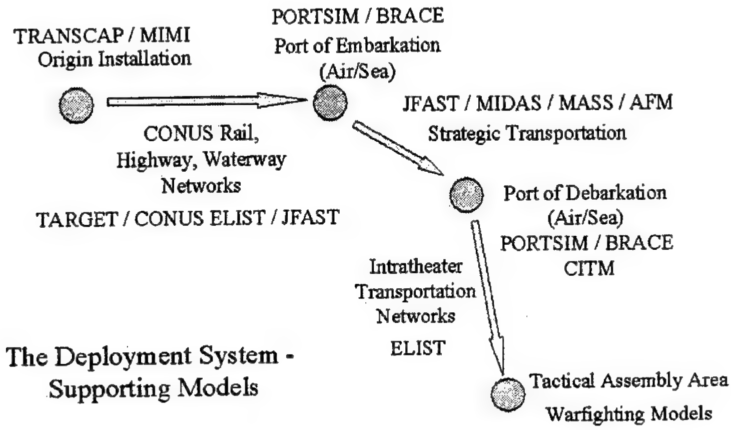


Figure 2 – Models and Simulations Used in Force Projection Analysis

The MTMCTEA portion of this deployment suite is grounded on detailed information on infrastructure and the individual equipment items that flow through the system. The Agency acquires and maintains infrastructure data on origin installations and depots and potential ports of embarkation and debarkation worldwide. These data are developed and deployed through advanced Geographical Information System (GIS) technology. MTMCTEA also constructs transportation facility and network data on possible theaters of operation, using GIS tools and a wide variety of data sources, including open literature, reconnaissance by Agency engineers, and intelligence community data. CONUS civil transport infrastructure data is supplied by the Department of Transportation (DoT) in the Federal Highway Administration (FHWA) National Highway Planning Network (NHPN), the Federal Railroad Administration's National Railroad Planning Network, an Inland Waterway database produced by a consortium of DoD and civil organization, and the FHWA National Bridge Inventory. The Agency was instrumental in the initial construction of the network bases in a series of cooperative ventures with DoT. These bases are enriched by "value-added" data generated by MTMCTEA. For example, the highway data include speed limits imputed by Agency analysts, and traffic congestion data are being developed by the University of Tennessee for incorporation into the NHPN. MTMCTEA adds the limiting characteristics of bridges and tunnels to the highway base through conflation with the National Bridge Inventory and through software for computing Military Load Classification for bridges, based on algorithms produced by the University of Maryland and Argonne National Laboratory.

Equipment detail is based on the DA Standard Equipment Characteristics File (ECF) for US Army items, which the Agency has maintained for Forces Command for many years. The database contains information on equipment for US Navy Construction Battalions and for selected Air Force items as well. A separate base of US Marine Corps equipment characteristics data, which the Agency has long maintained for USMC, is now being merged with the ECF to form a joint database. The coupling of equipment characteristics with the unit movements captured in the TPFDD is the business of MTMCTEA's Transportability Analysis Reports Generator or **TARGET** system. This is a group of models and programs that provide the capability to detail unit movement requirements at the line item number (LIN) level of detail. The system uses the ECF and various sources of unit equipment authorization data, including the DA Tables of Organization and Equipment (TOE), Modification TOEs (or MTOE data), and Unit Identification Code (UIC) equipment holdings from the Total Army Equipment Distribution Plan (TAEDP) system. TARGET uses the ORACLE database management system to merge the ECF data with specified units and create Unit Equipment Tables that drive the main analytical subsystems of TARGET. These are programs that load compatible unit cargo on organic vehicles, develop transport equipment requirements for multiple modes by detailed loading of equipment items on highway and rail assets, and perform aircraft loading. TARGET is often used in a standalone capacity to analyze the impact of future weapon systems changes, changes in the DTS, and changes in force structure on the number and kind of assets required to deploy US forces. More frequently, it provides the basic movement requirement for a TPFDD flow to the models of the FPM suite.

To understand the overall structure of FPM, it is useful to trace the models in "deployment" order, beginning with the installation. **TRANSCAP** is a discrete event, stochastic, constructive simulation of installation transportation operations at the Line Item Number (LIN) level of detail. It will have the ability to compare computed capabilities with outloading requirements. Where requirements exceed capabilities, it will recommend courses of action to eliminate transportation system deficiencies and provide estimated costs, automating the transportation system capability study process by which MTMCTEA evaluates the ability of a facility to support its wartime mission. It will calculate the numbers of railcars, trucks, containers and aircraft that can deploy from an installation over a given time period. The system will eventually consist of four modules:

- The first module developed was the Unit-Move Installation Module. It computes time-phased outloading capabilities for all transportation modes at unit move installations.
- The Multiple Installations/Depots Module will provide a tool to perform system-wide transportation analyses. It will permit a view of installations and depots operating concurrently to better allocate resources among competing facilities.
- The Reception, Staging, Onward Movement and Integration (RSO&I) Module will compute reception capability within the destination theater's staging and intermediate bases and Tactical Assembly Areas.
- The Sustainment Module will automate the transportation system capability study process for depots and ammunition plants.

TRANSCAP is currently under development as a joint effort of MTMCTEA and Argonne National Laboratory.

The **MIMI** model is US Forces Command's force generation model. It analyzes movement of units to mobilization stations, their subsequent equipping (including cross-leveling) and training, encompassing allocation of physical resources such as firing ranges.

ELIST is a discrete event simulation system that was first developed to determine the adequacy of transportation logistics for the theater portion of a course of action. The system executes a series of movement requirements over a constrained theater network using constrained transportation assets. ELIST recognizes military lift assets arriving in theater and performs unit marry-up of equipment and personnel. Thanks to the generality of its routines, ELIST also provides the option of simulating the CONUS portion of a course of action. With its ability to translate road and rail infrastructure networks from GIS databases, ELIST can load a CONUS network and flow movement requirements from points of origin to POEs. ELIST addresses the question of whether transportation infrastructure and lift allocations are adequate to support the movements of specified force structures and supplies to their destinations on time.

ELIST is a mature tool developed from an Argonne National Laboratory (ANL) advanced prototype of

1993, called the Logistics Intratheater Support Tool. ELIST has been used in many exercises and CINC support analyses, including applications in Korea and Bosnia. The model is a component of the standard US TRANSCOM Analysis of Mobility Platform (AMP). MTMCTEA and ANL recently completed release 7.2.2 of the software.

PORTSIM is a discrete event, stochastic simulation of ocean terminal operations that determines port throughput, identifies constraints, and calculates port clearance times for deploying units. Upon completion, PORTSIM will serve as an agency, service, and CINC analytical, planning, training, and execution tool. PORTSIM is a fully functional simulation currently in use for seaport evaluation and war planning support. It is not yet integrated with other components of the FPM suite, but the system is the first of MTMCTEA's models—and one of the first in the Department of Defense—to become HLA-compliant. The functionality of the system is still under active development by ANL and MTMCTEA, with additional analytical and development support from the Virginia Modeling, Analysis and Simulation Center (VMASC). VMASC is currently working to restructure the port of debarkation functionality of PORTSIM. PORTSIM development began in June 1994. The initial operational capability (IOC) was delivered December 1996. Near-term development objectives include:

- Connectivity and interaction with a scheduling model.
- Basic linkage and connectivity with MIDAS (GRCI).I.
- Implementation of helicopter processing.
- Receipt processing pulled from staging.
- Implementation of TRANSCAP/TARGET/PORTSIM Rail Module.

The **BRACE** model noted in Figure 2 is an aerial port simulation, capable of representing either the port of embarkation or debarkation. It is a tool of the Air Mobility Command.

The intertheater or strategic lift portion of the deployment is handled by one of several selectable standard models, including the TRANSCOM **JFAST** model. JFAST is a high speed analytical tool used for making detailed estimates of the resources required to transport military forces under various scenarios. It is a USTRANSCOM application originally developed by Oak Ridge National Laboratory. The other models noted are standard tools of long standing, MIDAS being the primary programmatic analysis tool used at the OSD level, and MASS and AFM airlift scheduling and simulation models.

PORTSIM and BRACE are also used to analyze the capability and workload requirements for the ports of debarkation in theater. **CITM**, or the Coastal Integrated Throughput Model, is under preparation by the Waterways Experiment Station of the Corps of Engineers. It will provide a PORTSIM-like simulation analysis capability for logistics over the shore operations and those in degraded ports (for example, those whose characteristics have been adversely modified by combat activities). It is expected that CITM will be available as a PORTSIM module.

Ensuring that all these models and data are smoothly and seamlessly integrated to provide the very best answers to deployment capability questions is MTMCTEA's ongoing challenge. Successes to date provide every confidence that we will reach our goals, supporting the deployment community with focussed analysis based on the best tools of modern technology. Our objective: a Defense Transportation System capable of projecting U.S. military power rapidly and responsively to any point on the globe where it may be needed.

Command and Control Systems Supporting Force Deployment

Development, testing and fielding of software for operational support is not in the MTMCTEA mission. Figure 3 below shows a shower of command and control systems employed today or in development for tomorrow, all of which have a direct impact on the deployment process. The figure indicates which portion or portions of the flow are monitored or controlled by these systems. A major analytical shortcoming is that no deployment model or suite available today includes the impact of the success or failure of these control systems as a component of the overall system simulation. Given the absolute importance of these control mechanisms, it is unfortunate that we have little, if any, concept of what communications and coordination failures would mean operationally. That there is in fact a perceived danger is clear from the investment the Department of Defense is putting into infrastructure and information protection.

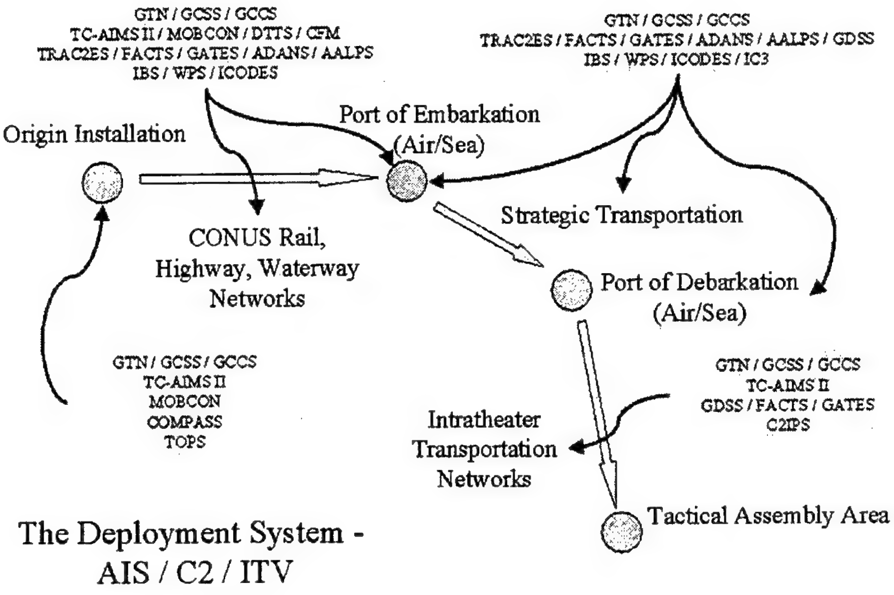


Figure 3 – Automated Information System, Command and Control, and In-transit Visibility Systems in Deployment.

Network Data Available for Deployability Analysis

For the Continental United States, there exists an extremely rich supply of Geographic Information Systems oriented data on the location and characteristics of the major transport networks: road, rail and waterway. For example, Figure 4 shows the Bureau of Transportation Statistics (BTS) version of the National Highway Planning Network (NHPN), the basic highway framework database for the United States. This 400,000-mile network is maintained by the Federal Highway Administration of the Department of Transportation, and consists of locational and attribute data on Federal Aid Highways in the 50 States and Puerto Rico. Currently in release version 3.0, the base is available for free download from the BTS. The geographic accuracy of the NHPN is roughly equivalent to map scale 1:100,000. The somewhat sparse attribute set of the NHPN is enhanced substantially through use of the DoT's Highway Performance Monitoring System, which includes a universal dataset of characteristics maintained on all reportable highways, and which is consistently geolocated with the NHPN. Other elements of the HPMS are reported only for sample data; expanding these to characteristics of even contiguous highway segments is quite dangerous.

The National Railroad Planning Network, also at a nominal scale of 1:100,000, is a network database of all railway mainlines, railroad yards, and major sidings in the continental U.S. It is maintained by the Federal Railroad Administration. A less detailed version, at a nominal scale of 1:2,000,000, is also available. These files are also downloadable from the BTS site.

The National Waterway Network is a network database of all navigable inland and intracoastal waterways, Gulf, Great Lakes and coastal sea lanes, and major sea lanes between the continental U.S., Alaska, Hawaii and Puerto Rico. Maintained by the US Army Corps of Engineers, it is the product of a broad consortium of transportation organizations, academic and commercial institutions.

The National Bridge Inventory (NBI), maintained by the Federal Highway Administration with input from the individual States, is a record of the characteristics and condition of some 600,000 load-bearing structures in the United States. A major effort of MTMCTEA geographical analysts was the conflation of this database with the NHPN to provide a network capacitated by the limiting features of bridges and culverts. Recently, the Federal databases have been oriented to a linear referencing (milepost) system which has improved the geographical association of the network and structures (though possibly at the cost of accurate location of the structures themselves). In addition to linking these databases, MTMCTEA analysts supported by Argonne National Laboratory, the University of Maryland, and the Corps of Engineers have developed algorithms for calculating Military Load Classifications for NBI structures, permitting evaluation of the ability of highway links to pass outsize and overweight loads such as Heavy Equipment Transporters moving M-1 tanks.

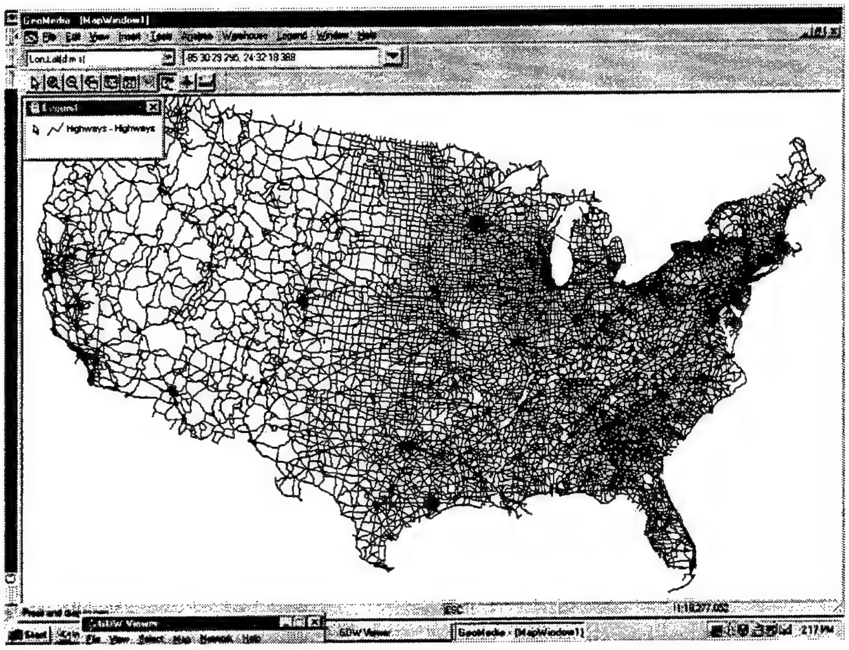


Figure 4 – The National Highway Planning Network

Since the speed with which military movements can be made through the highway network are strongly dependent on the level of congestion in the net, MTMCTEA has supported research by the University of Tennessee and others on predicting congestion reasonably. Existing methods involve either very large travel requirements/desires datasets, or depend on very large simulations, such as the Department of Energy's TRANSIM model. Since time in war planning is generally very short, elaborate methods of this sort are not normally available. Consequentially, research has focussed on generating simple profiles with which average daily traffic can be allocated to peaks and valleys of density over time. Figure 5 shows an example of the type of curve, based on HPMS data, the UT researchers developed. The final travel speed on a link derived from this distribution is adjusted, depending on such factors as lane width, presence of trucks (including the traffic being scheduled), the number of intersections and traffic lights on the link, and similar location-specific data. This methodology is relatively crude, but even at its current state permits generating more information than we are yet able to employ well in deriving optimal routings, though we now have much improved travel time algorithms.

The question of accuracy and precision in the Federal databases has always been important. Because the purposes of the users of these National database are various, their requirements for locational precision are also various. Current efforts at MTMCTEA include provision to the installation and division transportation officers of real-time condition and capacity data on the routes leading from major deployment origins (the so-called Power Projection Platforms) and their assigned ports of embarkation. Data are being acquired

$$V = AADT * K * D * 1/PHF$$

Data elements from HPMS

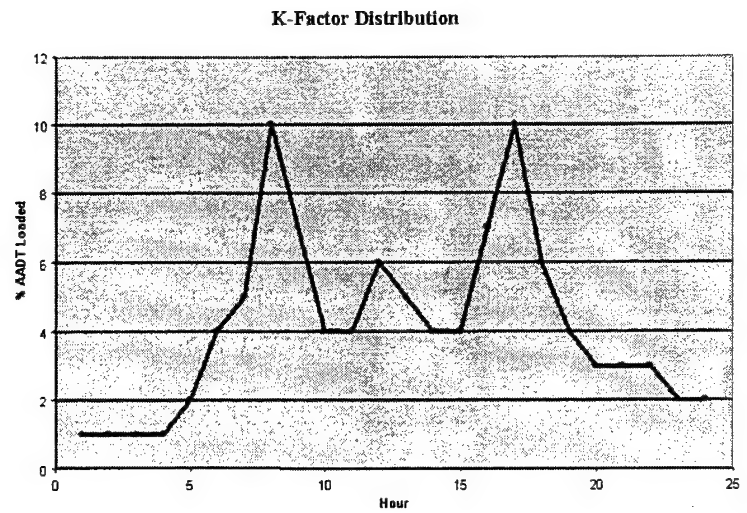


Figure 5 – Example of a Congestion Factor Distribution.

from sensors and reporting systems on the Worldwide Web, collated and presented so that movement planners can be aware of weather, construction, congestion and other time-dependent aspects of the route. This system exists today in prototype, with many of the data-gathering systems in place. Because the locational accuracy of such items as traffic cameras is very high, it is necessary for the underlying network data to be of equal precision and accuracy. The scale and precision of the national bases, particularly of highway and bridge data, is not adequate for this application, and MTMCTEA is being forced by the level of the requirement to purchase high-precision navigational data prepared by commercial industry.

Information on transportation networks in potential theaters of operation is typically not available in any organized fashion, and MTMCTEA analysts build the networks essentially on the fly. It is not uncommon for a requirement for transportation infrastructure data in a developing country to be drawn from commercial maps published for tourism purposes. Of course, such sources are validated to the extent possible against National intelligence sources. Particularly important are photographic imagery from space or aerial platforms (most furnished by the National Imagery and Mapping Agency or NIMA). Attribute data are not typically derivable directly from this imagery, and we have resource to intelligence banks such as the Modernized Integrated Data Base (MIDB). The agency has made significant investments in the data acquisition and storage arena, using Government and commercial means of high bandwidth communication using satellites.

Outstanding Requirements

It should be quite clear that, even though deployability analysis is now supported by an almost embarrassing wealth of data of ever-increasing accuracy and detail, we have a substantial list of additional requirements. Our major shortcomings are now analytical, with perhaps the most pressing requirement being an appropriate algorithm for path planning. The problem is non-trivial: finding an optimal route through a network with time-dependent congestion, taking into account the impact on congestion of the scheduled traffic itself, including the influence on route selection of structural capabilities. And although the emphasis in recent years on acquiring an adequate fleet of roll-on/roll-off ocean vessels to support major deployments has reduced the military reliance on the commercial ocean fleet, the demands of our current deployability goals require us to consider the use of intermodalism to reduce handling and increase the size of the fleet available at any given time. This further increases the complexity of the analytical problem. We would be very happy with a good heuristic or simulation methodology.

Other analytical requirements include determining how traffic redistributes in response to network changes and disruptions, either from natural or from manmade causes, and what the impact on physical movement of less than perfect information flow would be. An additional and not minor point is that acquisition of this capability requires money—not necessarily easy to come by in today's constrained government spending environment.

The Advanced Logistics Program (ALP)

The Defense Advanced Research Projects Agency (DARPA), famous for the introduction of the Internet, is now completing a three-year project, the Advanced Logistics Program [Sharkey, 1997], the original goal of which was to permit:

- Automatic development of logistics plans in one hour.

- Minimal seaport staging and globally optimized lift scheduling.

- Automatic detection of plan deviations during execution, with replanning accomplished within 15 minutes of deviation detection.

- Continuous demand generation and sourcing against DoD and commercial inventories.

MTMC was particularly enchanted by the goal of minimizing the need for seaport staging through optimized arrival of cargo at the port, a high-precision flow control similar to the Just-In-Time thrust of much commercial flow operations. The ALP program envisioned greatly improved visualization of on-going operations and plan deviations. The program culminated in a series of technology demonstrations at DARPA's Technology Demonstration Center in February 2000 [Carrico 00]. The emphasis had shifted in the interim to demonstration of architectural concepts. The centerpiece is an Internet based Cluster Architecture to facilitate cooperative analysis, planning and replanning over large geographic areas and numerous organizations, both private and commercial. This architecture envisioned installing specific functionality (such as port workload prediction) through software "plug-in" technologies. Intelligent agents and sentinels were designed to monitor select execution critical points (or "pulse points") and initiate replanning based on the current values of penalty functions. While some plug-in components were UNIX based, the overall architecture was to be Windows NT and Java based. The demonstrations appear to be regarded as successful, suggesting that some deployment analysis requirements might be satisfied architecturally by the ALP structures. Follow-on work in technology development is being explored, through an ALP Integrated Process Team that will investigate funding for FY00 and 01 prototype, develop an action plan for ALP transition to implementation, identify and monitor technical and functional requirements used in the technology demonstrations, assess the intent of other Services and Agencies to implement ALP, and develop an ALP implementation strategy for the Army [Moore, 2000]. Unfortunately, the global optimization of lift scheduling remains a difficult and unachieved objective.

This is not to say that there has been no progress. Sandia and Los Alamos National Laboratories have developed extremely detailed simulations of large-scale transportation systems at the individual vehicle level (TRANSIMS) which have attracted considerable interest. Los Alamos successfully demonstrated a prototype military application of a similar technology in 1998 as part of a design effort for a National Transportation Network Analysis Capability in support of the Department of Transportation [LANL 98]. We are receiving assistance in enhancing our nodal simulations from commercial/academic consortia such as the Virginia Modeling, Analysis and Simulation Center. Argonne National Laboratory continues its efforts in support of our modeling and simulation efforts, and we are reasonably confident that the long term picture is bright.

References

Carrico, Todd M. "A Full Calendar for JLTO," Joint Logistics Technology Pipeline, No. 4, page 1, Feb 2000.

Los Alamos National Laboratory, National Transportation Network Analysis Capability (NTNAC), Dec 1998.

Moore, Miranda. "Advanced Logistics Program Integrated Process Team Update," briefing to ALP General Officer Steering Group, 24 April 2000.

Military Traffic Management Command Transportation Engineering Agency (MTMCTEA). MTMCTEA Regulation 10-1, Organization and Functions – Organization Mission and Functions. Feb 2000.

Sharkey, Brian, DARPA Program Manager, and Atkinson, Cola. Briefing, "1997 Advanced Logistics Program Demonstration," 18 Sep 1997.

GEN Eric Shinseki, US Army Chief of Staff, Speech to Association of the US Army, 12 Oct 99.

Section 2

Information Flow & Exchange

A Collaborative Approach for Dynamic Battle Control and Force Level Assessment in Theater Air Operations.

Michael Neyland
Science Applications International Corporation
European Programs Division
Einsiedlerhof, Germany
Michael.Neyland@saic.com

Kevin Koehn
The Warrior Preparation Center
Einsiedlerhof, Germany
Kevin.Koehn@wpc.af.mil

Abstract

Joint Vision 2010 proposes the integration of new and emerging technologies with operational concepts to achieve full spectrum dominance throughout the range of military operations. This paper advocates a new direction for research on integrated planning, scheduling, and control within a large-scale enterprise. It presents a collaborative plan for employing the increasing capabilities of technology to collect, generate, and disseminate critical air campaign information to achieve timely responsiveness in the areas of dominant maneuver, precision engagement, and dynamic battle control. Additionally, this paper advocates employing the synergy of strategic and tactical modeling to analyze the potential effects of air campaign planning and force allocation decision processes prior to task order production.

1. Introduction: Dominant Maneuver, Precision Engagement, and Dynamic Battle Control

Within the framework of future Joint Operations, Joint Vision 2010 defines dominant maneuver as 'the simultaneous application of combat power throughout the battlespace' [1], and precision engagement is described as the 'capability to precisely apply effects and/or forces to achieve ... operational results'. [2]

The context of this paper is centric to the capability of the Air Component Commander to simultaneously apply combat air power throughout the battlespace and the ability to precisely apply air power to achieve desired operational results.

Dominant maneuver from the perspective of the application of air power throughout the theater of operations battlespace is dependent upon the integrated planning, scheduling, and control of broadly distributed resources with multiple semi-autonomous participants, i.e., Joint and Coalition Air Forces. The effective application of dominant maneuver may be achieved via a collaborative technique of employing the increased capabilities of technology to collect, generate, and disseminate critical air campaign information to support the processes of planning, scheduling, controlling, analysis, and assessment of combined air and space assets.

The effective application of air power for precision engagement is simiarily dependent upon a collaborative technique of employing technology advances in real-time information management to re-direct air power for dynamic battle control. For the purposes of this paper, Dynamic Battle Control (DBC) is defined as '... a tightly closed loop process that enables the vigorous and rapid detection, identification, prioritization, sequencing, and attack/attack assessment of air and selected ground targets during the execution phase of air combat operations' [3].

This paper is presented from an 'operations' perspective and is intended to encourage the interaction of battle managers (practitioners) and engineers from the Electronic Systems Center, Air Force Research Labs, and Universities (theoreticians) to examine the concept of a collaborative environment for the effective and efficient application of air and space power supporting the Air Expeditionary Forces and combined air operations. From this operations viewpoint, this paper opens with a description of selected operational initiatives fielded by the United States Air Forces Europe (USAFE) to support air campaign planning, execution, and analysis and continues with a review of efforts to improve these initiatives to enhance dominant maneuver and precision engagement by employing fielded and planned systems in a collaborative environment. This paper also recommends a concept for incorporating modeling and simulations to support real-time operations.

2. The Integrated Targeting Environment (ITE) and Mission Analysis Tracking and Tabulation System (MATTS) – First Steps In Support of Dominant Maneuver and Precision Attack

Combat planning and unit level air campaign analysis during Operation ALLIED FORCE and the Air War Over Serbia (AWOS) included an initial capability to provide an automated system for targeting and weaponeering generation and a capability to generate analysis reports of strike aircraft and weapons performance against allocated targets. The automated targeting and weaponeering reports were designed by the Headquarters (HQ) United States Air Forces Europe (USAFE) Intelligence (IN) division under the ITE software initiative. The analysis reports were designed by an Operations Assessment Team (OAT) from the USAFE Warrior Preparation Center (WPC) under the MATTS rapid prototype effort.

2.1. The Integrated Targeting Environment (ITE)

ITE was implemented by HQ USAFE and the 32nd Air Intelligence Squadron (32nd AIS) to improve intelligence support to Operation ALLIED FORCE. A single web-based application was used to access contingency data, including an Electronic Target Folder (ETF) database with platform-independent, global-web access for all fixed target data. The ETFs contained over 1,000 executable target folders with imagery and over 4,000 cockpit video clips. After Operation ALLIED FORCE, HQ USAFE/IN examined the products used at each stage of the air campaign cycle, and designed ITE to facilitate the processes that produced them. The result was a set of three modules which formed a robust capability to support the strategic guidance, targeting, and weaponeering processes. [4]

The concept behind ITE was to provide user-friendly, web-based intelligence applications supporting the targeting cycle from target acquisition through mission execution. ITE is deployed as interoperable modules that support spiral development of additional functionality, utilizing existing databases when available. As such, the modules share information sources. As the tools are web-based, access to them is platform independent; only a web-browser and connectivity to the proper network is needed.

The modules provide applicable decision-makers a tool with which to assess and accurately strike selected high-value fixed target complexes with the most appropriate weapons. Additionally, these targeting products serve to assist the Air Operations Center (AOC) Master Air Attack Plan (MAAP) and unit level planners in Air Tasking Order (ATO) development and mission planning to ensure that they limit to the maximum extent possible unintended collateral damage.

ITE facilitates linking commander's guidance and objectives to target systems, initial analysis of the target system, analysis on restrike through incorporation of Battle Damage Assessment (BDA) reports and cockpit video. The web-based applications provide a platform for easy access and rapid dissemination of target materials to mission planners and units. [5] Figure 1 graphically depicts potential ITE support to the application of air power and shows the planning tools that may be used as building blocks for precision engagement.

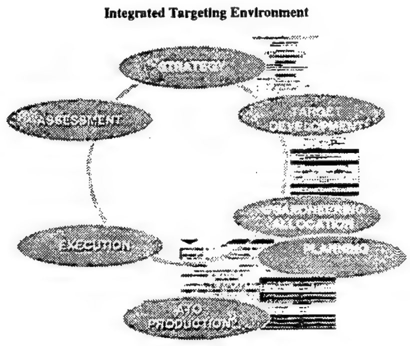


Figure 1 – ITE Building Blocks for the Application of Air Power

2.2 The Mission Analysis Tracking and Tabulation System (MATTS)

MATTS was initially designed to capture AWOS reporting data to support mission/sortie performance and weapons effectiveness assessments. As the air campaign continued, MATTS became an electronic archive of Mission Reports (MISREP) and imagery data for the air operation. MATTS evolved into a versatile database for analysis and reporting of the air operations during the AWOS. [6]

The initial AWOS MATTS products were aircraft, munitions, and target reports. During the AWOS, these reports permitted the tracking of multiple measures of effectiveness for analysis and reporting related to the following four questions:

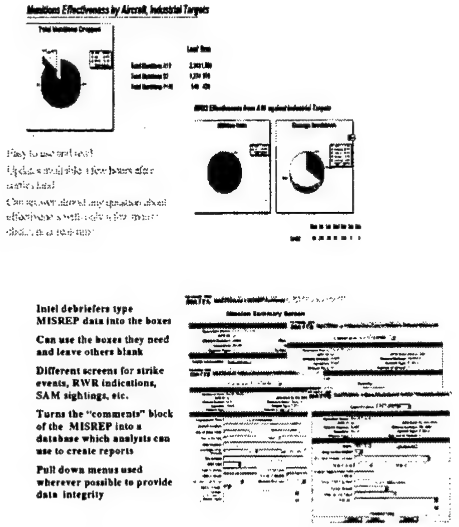
1) Did the aircraft fly? If not, why not?
2) Did the aircraft drop? If not,why not?
3) Did the munition hit the target? If not, why not? and
4) If the munition hit the target, what damage did it do? [7]

The significance of MATTS reporting and analysis capabilities culminated in a USAFE WPC effort to transition MATTS to a web-based, menu-driven system. The web-based system retained the original aircraft performance and weapons analysis and was expanded to provide an automated capability for the generation, collection, and dissemination of all theater MISREPs. The web-based MATTS also provides rapid access to current and post Air Tasking Order (ATO) data, BDA reports, imagery, and target data. MATTS products are accessed by operations and intelligence centers, wings, squadrons, and agencies via the United States (US) Secure Internet Protocol Router Network (SIPRNET).

Follow-on design and development work included an automated interface to the Theater Battle Management Core Systems (TBMCS) and the HQ USAFE ITE program to facilitate across-the-board improvements to information dissemination to support the application of air power throughout the battlespace and to precisely provide air power to the right targets at the right time.

Examples of MATTS products are shown in Figures 2. Figure 3 graphically depicts potential MATTS support to the application of air power throughout the theater battle space and shows the planning tools that may be used as building blocks for precision engagement.

Similar to ITE, the MATTS products are web-based and easily disseminated to theater and national agencies for battle planning and target analysis. This capability provides the basic building block for MATTS software tools to support application of air power throughout the theater battle space. Figure 4 graphically depicts the reporting and assessment support MATTS provides in the planning, execution, and assessment cycle of combat operations.



Easy to use and read

Updates available a few hours after sorties land

Intel debriefers type MISREP data into the boxes

Can use the boxes they need and leave others blank

Different screens for strike events, RWR indications, SAM sightings, etc.

Turns the "comments" block of the MISREP into a database which analysts can use to create reports

Pull down menus used wherever possible to provide data integrity

Figure 2 - MATTS Products - Charts and Reports

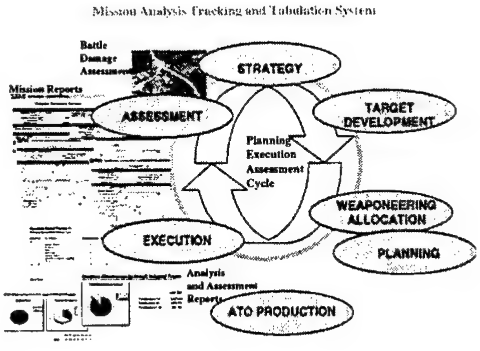


Figure 3– MATTS Support to the Air Campaign

As part of the web-based initiative, immediate design and development work included an automated interface to the HQ USAFE ITE program to facilitate across-the-board improvements to information dissemination for battle management and Dynamic Battle Control (DBC). In addition, development efforts addressed an automated interface to Theater Battle Management Core Systems (TBMCS) operations and target databases to enhance analysis and campaign assessment capabilities.

MATTS and ITE were separately developed, but mutually complementary to the planning, execution, and analysis of theater air operations. In early January 2000, the efforts of both development teams merged to support battle management DBC information dissemination in a theater of operations. Figure 4 graphically represents this collaborative approach between MATTS and ITE. This commonality of effort led to the concept of a collaborative Joint Air Campaign Tool (JACT) for a battle management information distribution system. The JACT concept reflects an emphasis on the collaborative use of numerous software programs to support all phases of air operations.

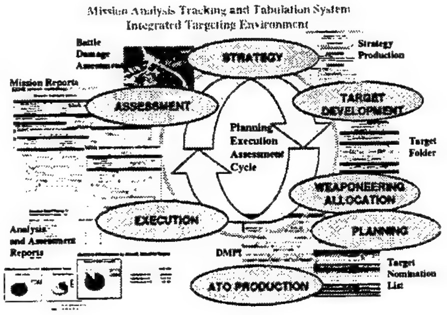


Figure 4 – Complementary Support to Battle Management and Dynamic Battle Control

3. The Joint Air Campaign Tool

Incorporating the web-based applications of MATTS and ITE as collaborative tools established the JACT baseline concept for supporting the multi-dimensional strategic and theater-level decision making process. The JACT concept represents the first step towards developing a system supporting dominant maneuver and dynamic battle control. The JACT modular concept facilitates integration and interoperability with Joint Services and Coalition forces command and control (C2) systems. JACT applications accelerate the dissemination and cross sharing of critical, time-sensitive battle management information.

A principle JACT objective is to take full advantage of the diversity of fielded, projected, and individual initiative battle management tools used in all phases of strategic and theàter-level air operations.

To accomplish this, JACT presents a collaborative architecture taking advantage of the diversity of air campaign tools and maximizes a distributive use of these tools without disturbing their original objectives.

JACT is intended as a "bridge" to other planning and decision aid tools. The basic principle is that nothing is wasted and nothing is ignored. The tools that are currently in use (whether designed/fielded by the centers (Electronic Systems Center), labs (Air Force Research Lab, Command and Control Battle Lab), or independently designed by an Airman for a specific purpose) are of value to at least one part of the planning, execution, reporting, and analysis cycle, and may be of value to the whole process. Accomplishing a cross-feed of databases and reports significantly improves the return on dollars invested in the design and fielding of planning and decision aid tools.

Figure 5 depicts the JACT concept. As shown in Figure 5, the intent is to integrate Component, Joint and Coalition battle management programs to develop a fully integrated 'system of systems' databases for air, ground, and sea operations. This series of integrated programs will encompass an eclectic group of software modules ranging from logistics to weather to dynamic re-targeting. The software modules include programs designed for individual service, Joint, and Coalition operations. The software modules which comprise JACT interoperate with each other to exchange common information for a wide range of battle management air and space activities. The end result is a dissemination of deployment, employment, execution, and assessment data throughout the command and control structures. The Joint Force Air Component Commander (JFACC) enterprise concept is to evolve JACT to a Dominant Maneuver / Dynamic Battle Control decision support tool. The original JACT concept focused on the efficient and effective application of force against targets. Fielding a JACT-based decision support tool expands from the forces to target focus and encompasses the end-to-end applications of Component, Joint, and Coalition air power from deployment to theater operations.

The modular, web-based development philosophy used to integrate the MATTS/ITE software modules supporting reach-back and forward-deployed operations.

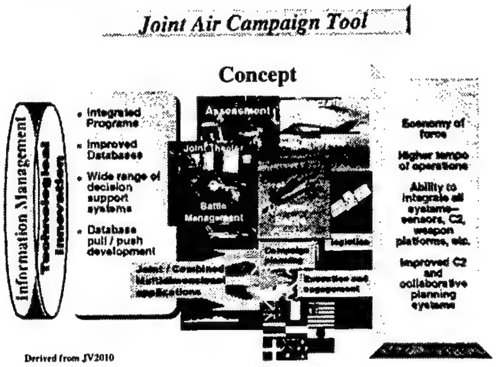

Joint Air Campaign Tool
Concept
Information Management
Technological Innovation
• Integrated Programs
• Improved Databases
• Wide range of decision support systems
• Database pull / push development
Assessment
Joint Theater
Battle Management
Logistics
Joint / Combined Multidimensional applications
Execution and engagement
Economy of force
Higher tempo of operations
Ability to integrate all systems—sensors, C2, weapon platforms, etc.
Improved C2 and collaborative planning systems
Derived from JV2010

Figure 5 – Joint Air Campaign Tool Concept Development

It should be noted that other services and Air Force Agencies have addressed the JACT concept. The Federated Assessment and Targeting Enhancements (FATE) initiative developed by the USAF Command and Control Battle Lab presents an excellent example of other JACT concepts. What makes the JACT approach unique is that JACT is based on the MATTS/ITE foundation of a firmly established, fielded, and tested program. Additionally, the JACT concept expands beyond the Air Component requirements to include the integration of Joint and Coalition battle management programs.

4. Modeling and Exercise Applications

Another JFACC enterprise concept is the application of modeling and simulation (M&S) tools to provide near real-time evaluations of strategy to task planning and subsequent Air Tasking Order generation. The concept is to incorporate M&S tools into the operations and intelligence real-world planning and execution environment. As the operations and intelligence assessment and planning process matures towards a production ATO, the JACT M&S tool will push the battle planning information to other M&S tools for "quick look" efficiency and effectiveness evaluations with an eye towards effects-based targeting. As a result of these evaluations, the production ATO may be modified to achieve a more effective use of allocated forces. The application of this concept is illustrated in Figure 6. The reverse concept may also be applied to modeling and exercise support. The 'lessons learned' and practical applications of real-world theater command and control operations may be assimilated into the exercise environment, such as Joint Theater level Simulation and Air Warfare Simulation Model, via the databases resident within JACT. Figure 6 presents a conceptual view of air campaign and modeling tools interaction.

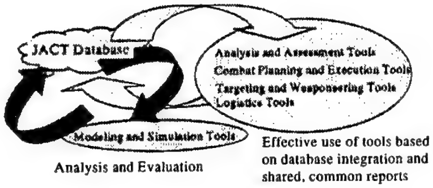

JACT Database
Analysis and Assessment Tools
Combat Planning and Execution Tools
Targeting and Weaponeering Tools
Logistics Tools
Modeling and Simulation Tools
Analysis and Evaluation
Effective use of tools based on database integration and shared, common reports

Figure 6 – Air Campaign and Modeling Tools Integration

5. Decision Support Tools - A Final Note

For the purpose of this paper, information dissemination is considered the interaction of relevant information from various battle management software modules to aid battle managers in the deployment processes and theater planning, execution, analysis, assessment, and reporting processes. Decision Support Tools are defined as the statistical, Monte Carlo, Deterministic, and/or model predictive control programs which support, to an extent, the development of recommended courses of action based on information derived from the battle management programs. The intent is to provide the Component, Joint, and Coalition Commanders with suggested courses of action for deployment or theater operations. From a force on force projection, the Decision Support Tool is intended to encompass the availability of air, ground, and sea forces in a combat theater for commitment to air operations. This paper is not intended to discuss the variety of decision support tools, other than indicate that future developments of the JACT model will incorporate these programs to advance Component, Joint and Coalition operations.

6. Abstract Summary

The "enterprise premise" is to employ a collaborative approach incorporating the capabilities of separate and distinct fielded and prototyped tool sets to achieve revolutionary advances in the full spectrum dominance of warfighting capabilities. The multi-dimensional applications of battle management and modeling concepts presented in this paper enhance dominant maneuver and dynamic battle control in the operations environment and support improvements in the exercise and simulation environment. Incorporating a collaborative integration and interoperability approach takes full advantage of the diversity of campaign tools, and maximizes the distributive use of these tools for Component, Joint, and Coalition operations.

Acknowledgment

Capt. Kevin Moffatt, USAF, HQ USAFE/IN, provided key insight into the operational concepts and integration of ITE with MATTS and other command and control systems. The authors are indebted to his willingness to provide background material and briefings to support this paper.

Bibliography

[1] "Concept for Future Joint Operations", Joint Warfighting Center, Fenwick Road, Fort Monroe, VA, para 6-12, p.50, May 1997.

[2] "Concept for Future Joint Operations", Joint Warfighting Center, Fenwick Road, Fort Monroe, VA, para 6-15, p.51, May 1997.

[3] "Concept of Operations for Dynamic Battle Control (CONOPS) in JEFX-2000" Planning Draft, Command and Control Training and Innovation Group, Hurlburt Field, Florida, para II Definitions, p. 4, 15 Mar 2000.

[4] Capt. Kevin Moffatt, USAFE/INXY, "Integrated Targeting Environment", Overview Presentation, Mar 00.

[5] Capt. Kevin Moffatt, USAFE/INXY, "USAFE Integrated Environment - Point Paper, 19 Jun 00.

[6] Joe J. Puckett, "Operational Assessment Team's Analytical Support for the Kosovo Air Campaign", *Phalanx - The Bulletin of Military Operations Research*, Vol 32, No. 3, Alexandria, VA, Pp 1,32, Sept. 99.

[7] Joe J. Puckett, "Operational Assessment Team's Analytical Support for the Kosovo Air Campaign", *Phalanx - The Bulletin of Military Operations Research*, Vol 32, No. 3, Alexandria, VA, p. 32, Sept. 99.

Autocatalytic Decision Overload in Command and Control Systems

Alexander Kott, Paul Hubbard, and Michael Martin
Logica Carnegie Group, Inc.
Pittsburgh, PA
{kotta, hubbardp, martinm}@logica.com

Abstract

We argue that the phenomenon of autocatalytic decision overload (ADO) may be a key mechanism contributing to the cases when a command and control system collapses catastrophically. We introduce the concept of ADO; describe a number of typical scenarios of such a collapse; propose a practical method for modeling the ADO phenomena; describe several computational experiments in which different aspects of ADO are simulated, and provide recommendations for architecting C2 systems in a manner that reduces susceptibility to ADO collapses.

1. Introduction

We view a C2 system as a network of decision-making entities (individuals, teams of individuals, and information-processing tools, i.e., artificial agents) operating jointly in accordance with organizational procedures and protocols. C2 systems acquire, transform, generate and disseminate information in order to acquire, allocate, and deploy enterprise resources so that enterprise objectives can be accomplished efficiently and effectively. A C2 system's ultimate product is the commands it issues to those elements of the Enterprise that execute direct effects on the environment of the enterprise. We call the totality of these executing elements – the Field. (In the control-theoretic literature, the term Plant is commonly used. We prefer to use the term Field in order to stress the generality and complexity of the enterprises.) The Field may include salespeople who are trying to affect the behavior of the buyers in the market; workers who assemble the products; trading floor clerks who execute the transactions; pilots of military aircraft who fly to bomb their targets.

Kott and Krogh [1] described and classified a number of undesirable, pathological phenomena that may occur in command and control systems of complex enterprises specifically because of the large, distributed nature of enterprise control systems, and despite the fact that the individual decision-makers are functioning as they were designed to function. In the present work, we focus on analysis of a pathology that belongs to the Capacity Saturation class of pathologies defined in the above-mentioned paper. The main feature of this pathology is that the C2 system, either internally or in interaction with the Field and environment, enters into a self-reinforcing cycle of increasing it own decision workload until the demand for decision-making exceeds the capacity of the C2 system. We suggest a name for this subclass of C2 pathologies – Autocatalytic Decision Overload (ADO). We hypothesize that ADO is common in practical cases of C2 collapses, and may be one of their main culprits.

2. Scenarios of Autocatalytic Decision Overload

To illustrate how common the contribution of ADO is to major failures of enterprise C2 – in both commercial and military enterprises – we offer the following set of typical scenarios.

Scenario A. After a major financial loss, a corporation forces one of its underperforming divisions into a major restructuring. Several new senior managers and advisors are brought into the divisional operations. The existing personnel dedicates a large fraction of their time to explaining and justifying their decisions to the new brass, modifying their procedures and plans according to the new guidance. The day-to-day decisions receive less attention and their quality suffers. Mistakes are made more often. The field force resents the erroneous guidance, and morale and discipline decline. Performance of the division suffers even further. The corporate management decides to step up the restructuring...and the vicious cycle continues.

Scenario B. A corporation faces a new, unexpected tactic employed by its competitor. The tactic is successful and rapidly makes the business plans and procedures of the corporation inapplicable. Management attempts to introduce new ideas and approaches. Field personnel are bewildered and call for explanations and support. Decisions with new unfamiliar approaches become harder just as the management attention is dissipated more than ever. Quality of decisions deteriorates. Management's confidence plummets and decisions take even more effort. The competition exploits the errors and continues to succeed in altering the market position, which in turn requires more adjustments in corporate business tactic... which in turn causes more confusion and errors...

Scenario C. Two corporate divisions coordinate their actions in order to utilize shared resources and to cooperate on joint business tasks. Division A recommends a particular action in order to deal with a customer order. Division B sees it as an error and disagrees. Division A argues that its recommendation is correct and meanwhile submits a request for another action to deal with another customer. The volume of communications begins to snowball... The management of both divisions is overloaded and begins to make more mistakes. This leads to more arguments and so on, until both divisions are engaged primarily in an exchange of arguments, generation of counter-arguments, mutual bickering and apportionment of blame.

The pure forms of the above scenarios may be difficult to discern in real-world situations where multiple factors and causal relations interact and obscure the picture. However, features of these scenarios can be compared to some of the near-disastrous events in C2 of the Israeli Southern Command in 1973 Arab-Israeli war [2]), or elements in the French command behavior in May-June 1940 [3]), or in the interactions within the buyout team of RJR Nabisco described in [4].)

As suggested by these examples, ADO could be responsible for a large fraction, and perhaps even the majority, of the ways in which a complex C2 system can collapse in a catastrophic failure. A quantitative measure of the extent to which ADO is prevalent in the real world could be produced by a comprehensive analysis of historic cases of catastrophic failures in military and corporate C2. This is a valuable direction for future research.

3. The Accuracy-Workload Tradeoff

A major factor in ADO is the fundamentally non-linear nature of the decision-making: the accuracy of the human decision-making drops as the decision workload increases (Figure 1). Experimental work reported in human-decision literature (e.g., [5]) generally agrees on an S-shaped function of accuracy vs. the workload. Louvet et. al. [6] report somewhat similar results. In the case of team decision-making, conclusive evidence is scarce, but one may conjecture that the S-shape tradeoff probably also applies. For artificial agents, we are not aware of research results, but one can suggest that a computer mechanism exhibits a flat accuracy level until the load reaches the limit of the processing capacity. Then, the accuracy drops linearly as more and more requests for decisions have to be left unanswered (i.e., implicitly answered by a random guess). Thus, an artificial agent within the decision-making network also exhibits a comparable non-linearity.

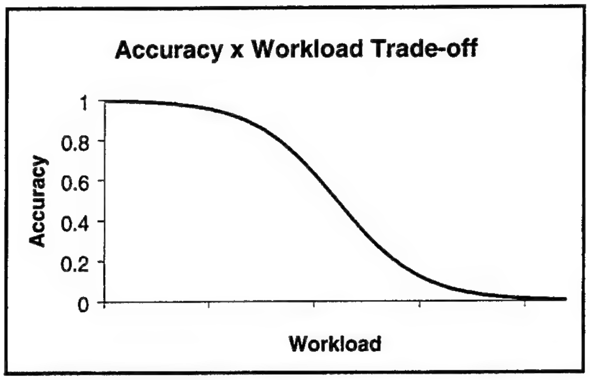


Figure 1. The tradeoff between the decision workload and the accuracy of the decision, as exhibited by human decision-makers.

The mere number of requests for decision is only one measure of the decision workload. Other factors contributing to a greater workload may include: complexity of the decisions; criticality or risk associated with the decision; uncertainty in the available data; latency of the available data. Impact of these factors should be accounted for, although for the purposes of this paper we focus on the simplest measure – rate of the decision requests per unit time.

As evidenced in the scenarios we described above, there also other aspects of the non-linearity in decision-making behavior. Of particular import seems to be the fact that a history of recent erroneous decisions or unexplained failures can impact the confidence of the decision-making entity and cause it to spend greater time searching for improved decisions. This also has to be left outside the scope of this paper. Here we focus on how the increased workload in decision-making entities can be self-reinforcing due to the drop-off in quality of the decisions.

4. Model of ADO

Consider the model that consists of two components, the C2 System and the Field organization. The C2 System component receives an input flow of orders from a higher authority (*u* – the number of orders per unit time) as well as a flow of requests for decisions from the Field component (*x2*). The C2 System produces a flow of commands (*x1*) and sends them to the field. Some of the commands are erroneous. A workload-accuracy trade-off function *f(x)* - approximation of the S-curve discussed earlier - governs the fraction of the errors. If the command is correct, the Field executes it successfully. An erroneous command results in a number of problems in the field. The problems manifest themselves in a number of requests for decisions generated by the Field and sent back to the C2 System. A constant coefficient *K* relates the number of erroneous commands and the number of new decisions that must be made as results of the errors. A greater value of *K* corresponds to a greater confusion caused by an erroneous command within the Field organization, and to a greater ability of the adversary to exploit the error.

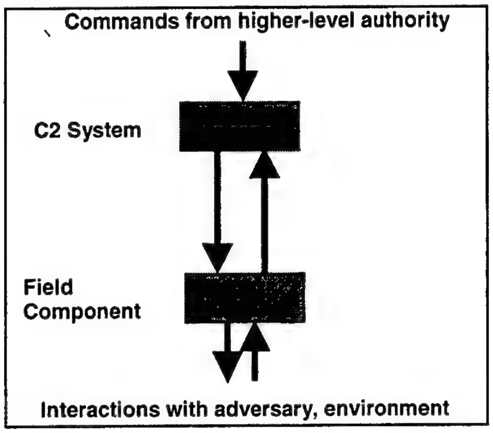

Commands from higher-level authority
C2 System
Field Component
Interactions with adversary, environment

Figure2 The system consisting of the C2 component and the Field organization.

If we use *x1* and *x2* as internals states, the dynamics of the system are given by

$$x_1(k+1) = x_2(k) + u(k)$$
$$x_2(k+1) = x_1(k) \cdot K \cdot f(x_1(k))$$

where

$$f(x) = 1 - \frac{1}{1 + e^{(x-a)/b}}$$

Linearization yields a sufficient condition for stability of the original nonlinear system: $K < 1$ or

$$-1 < [1 - \frac{(1 - x_1/b) \cdot e^{(x_1-a)/b} + 1}{(1 + e^{(x_1-a)/b})^2}] \cdot K < 1 \qquad (1)$$

and equilibrium points $\overline{x}_1, \overline{x}_2, \overline{u}$ must satisfy

$$\overline{u} = \overline{x}_1 \cdot (1 - K \cdot f(\overline{x}_1)) \qquad (2)$$
$$\overline{x}_2 = \overline{x}_1 \cdot K \cdot f(\overline{x}_1)$$

Results

Although there does not seem to be a closed form solution for the stability region, numerical computations using Matlab [7] yield the results depicted in Figure 3. The lines marked "20%" and "80%" show where the fraction of the erroneous commands issued by the C2 System stays at the levels of .2 and .8 respectively. We observe that

- although at *K*<1 the system remains theoretically stable, the higher values of *u* can lead to a rapid increase in the fraction of the erroneous decisions made by the C2 System; in essence this is a domain-specific type of instability;
- at *K* >1, the system can exhibit unstable behavior at higher values of *u*;
- as *K* increases (corresponding in the domain-specific terms to lower ability of the field organization to correct for the erroneous commands from above, and to a greater ability of the adversary to exploit the errors), the instability occurs at a progressively lower values of *u*.

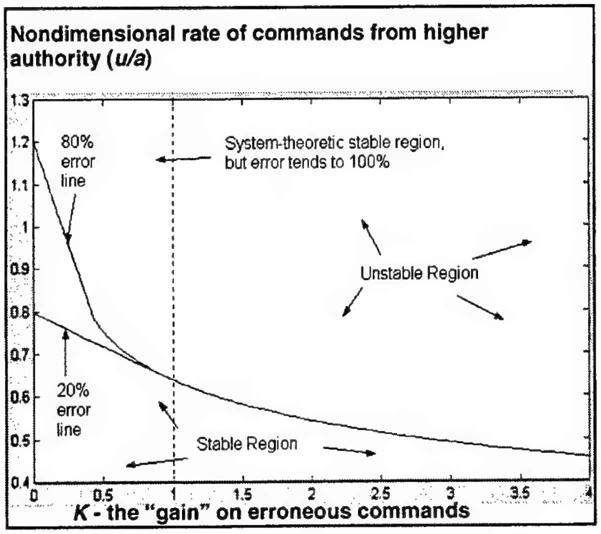


Figure 3. The stability region and the lines of constant error fraction.

5. Model of ADO in Stochastic Discrete-Event Systems

To explore the details of the behavior of the system described above at the edge of instability, we constructed a stochastic discrete-event model using the Simul8 tool [8]. Each command or request for decision was simulated as an event. Some of the commands generated by the C2 System were erroneous; this was determined stochastically with the average behavior similar to the workload-accuracy tradeoff discussed earlier.

Results

Figures 4 and 5 depict results of a typical experiment. As the flow rate of input instructions to C2 system increases, the accuracy of the commands issued by the C2 system decreases, until at some point the system enters an autocatalytic zone, the rate of commands and problems rises rapidly in an unstable fashion, and the system collapses.

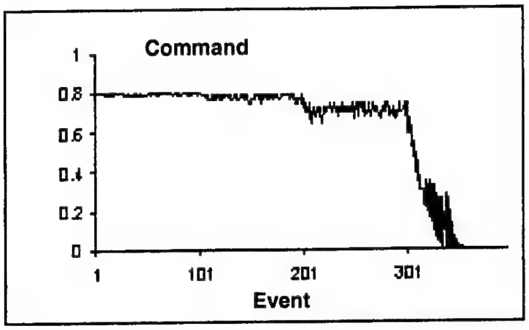


Figure 4. As the rate of demands from the higher-level authority is gradually increased, the C2 System exhibits only a minor degradation in performance (fraction of accurate commands generated) until the collapse point is reached where the C2 System's rate of correct commands drops to near zero.

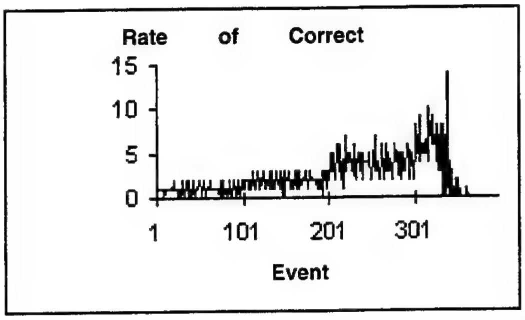


Figure 5. As the rate of demands from the higher-level authority is gradually increased, the C2 System is able to respond with increased rate of correct commands to the Field, until it reaches the point of collapse.

Different types of escalations in operational tempo had an impact on system stability. In a series of experiments, we increased the flow rate of the input instructions for the C2 system and observed the edge of collapse. The onset of collapse arrived at different flow rates of input depending on the manner in which the increase was affected. Generally, the onset of collapse came sooner if the flow rate of input messages was increased more rapidly.

The assumed level of accuracy of the C2 system under optimal conditions has significant impact on sensitivity to workload. A C2 system that operates at a 90% accuracy level under optimal workload conditions can handle a more rapid operations tempo (defined in terms of command flow rate) than can a system that operates at 80% accuracy during optimal workload.

Autocatalytic collapses were preceded in most cases by a lag in the conditions that precipitate it, almost always an increase in variability, and a decrease in mean performance accuracy. These may be exploited as diagnostics for upcoming collapses.

6. Model of ADO in Systems with Coordination between Decision-Makers

In this series of experiments, somewhat reminiscent of Scenario C described in section 2, we modeled a supervisory organization with two subordinate organizations (Figure 6). The subordinate components perform according to the workload-accuracy tradeoff. The supervisory component reviews the recommendations of the subordinates and detects errors with perfect accuracy. Errors are returned to the subordinate who committed them (i.e, their source) for re-work. This results in a self-reinforcing feedback loop that exacerbates the impact of workload-accuracy tradeoff on subordinates' performance.

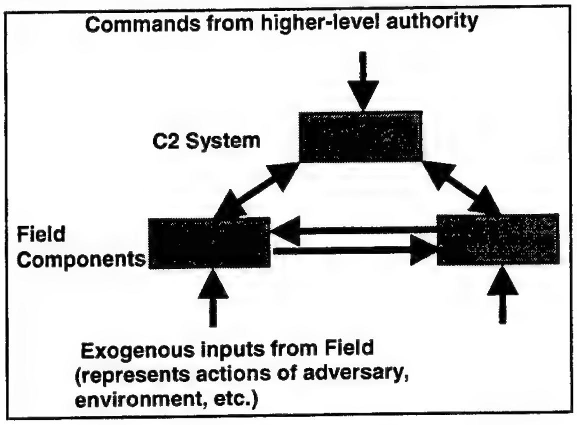

Commands from higher-level authority
C2 System
Field
Components
Exogenous inputs from Field
(represents actions of adversary,
environment, etc.)

Figure 6. The system involves coordination between two subordinate organizations.

When the subordinates are required to coordinate their decisions, it is performed in the following fashion. Messages received from a sibling component are checked for error, i.e., the acceptability to the receiving component. Error-detection capability is perfect. If an error is found, it will be returned to the source of the error (i.e., the sibling), after it has been processed. Processing means that it is subjected to the workload-accuracy tradeoff function of the component that received it and detected the error. The basic idea is that negotiation between two components of equal authority consists of a series of errors committed by the components as they continue to exchange information in an effort to reach an acceptable compromise. The negotiation process ends when the message is "correct" in the eyes of the receiver — at which point, the message is submitted to the supervisor.

The series of experiments examined the impact of the degree of coordination requirements and training levels on the time until system collapse. Degree of coordination was manipulated between 9% and 15% of the total number of messages, in 1% increments. Training levels were manipulated between 90% and 70% accuracy under optimal workload conditions. In the first system tested, both subordinate components were initialized to perform at 90% accuracy (i.e., the 90:90 condition). The second system used subordinate components that both performed at 70% accuracy (i.e., 70:70). The third system used assumed that subordinate components embodied different levels of expertise such that one component operated at 90% accuracy during optimal workload and the other operated at 70% accuracy during optimal workload (i.e., 90:70). A different random number seed was used for each simulation run. Time to system collapse was arbitrarily defined as the point at which both subordinate units' performance dipped below and remained below 25% accuracy.

Results

Time to system collapse generally decreases as coordination requirements increase (Figure 7). Thus, all else being equal, system collapse occurs earlier for a system that requires 15% coordination than for one that requires 9%.

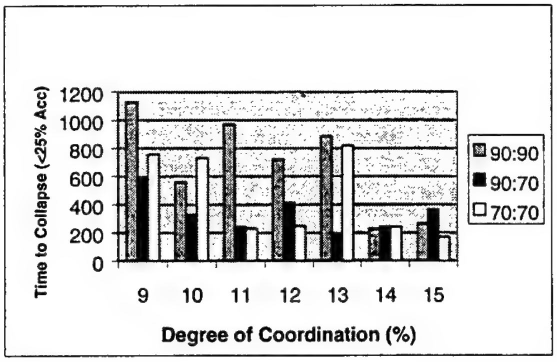


Figure 7. Increased coordination requirements lead to earlier onset of ADO.

Time to system collapse generally decreases as optimal accuracy level decreases. All else being equal, system collapse occurs earlier for systems comprised of components that perform at 70% accuracy under optimal workload than for those comprised of components that operate at 90% accuracy under optimal workload conditions. This statement seems to hold true for lower levels of required coordination. At higher levels of coordination (i.e., 14 & 15%), time to collapse appears to be similar regardless of training level. The system is incapable of handling these levels of coordination regardless of the performance each individual component is capable of under optimal conditions. Thus, optimal accuracy level interacts with degree of coordination required.

Time to system collapse is related to the weakest link in the system. The hypothesis is that a system comprised of components that perform heterogeneously under optimal workload conditions (i.e., 90:70) collapses at about the same time that a system comprised of components that all perform at lowest accuracy level (i.e., 70%). Indeed, in our experiments, time to system collapse for a 90:70 heterogeneous system seems to be significantly shorter than that of a 90:90 system, at least for low degrees of coordination. As for the comparison between the 90:70 system and the 70:70 system, results are less clear. Excluding coordination levels of 14 and 15% (the level at which the system seems to have been saturated), 4 of 5 pair-wise comparisons show the 90:70 system collapsing at about the same time as or before the 70:70 system.

7. Model of ADO in Hierarchical Systems

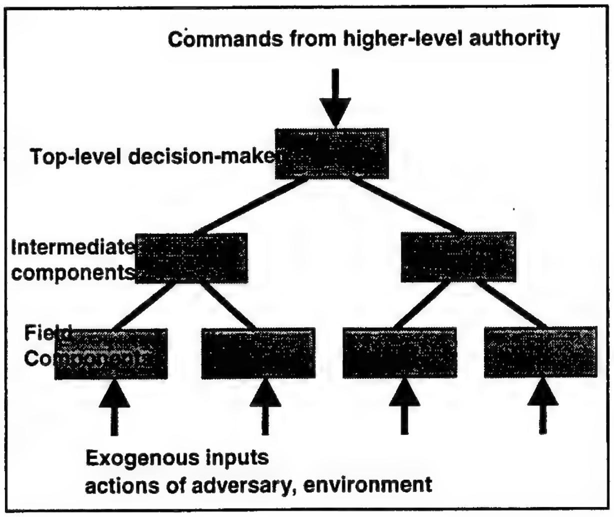


Figure 8. The hierarchical system.

In this series of computational experiments, we explored occurrences of ADO in hierarchical structures, including impact of weak performers, and differences in load from the field experienced by the performers. In the hierarchical model (Figure 8), intermediate components receive and interpret messages from their superior components and then direct messages to their subordinate components (as well as respond to their superiors). Each component introduces errors depending on the workload. At each location, the components generated decision demands toward the higher-level component as a proportional gain on erroneous commands received from above. The model allows for additional exogenous decision-demanding inputs directly from the field.

The stability of the models was analyzed subject to the rate of exogenous high-level inputs from above and the rate of exogenous inputs from the field. In section 4, two kinds of instability have been mentioned: namely, a systems-theoretic instability (i.e. bounded-input, bounded-input stability), and a domain-specific meaning of instability (i.e. production of 100% incorrect commands). In this experiment, systems-theoretic instability has been excluded with the use of saturation devices hence in the remaining sections, the instability referred to is the domain specific interpretation.

Results

Figure 9 shows the average error rate in commands to the field components as the exogenous input from above is increased. Similarly to Figure 4, the system maintains its performance at a near-constant level and then rapidly collapses.

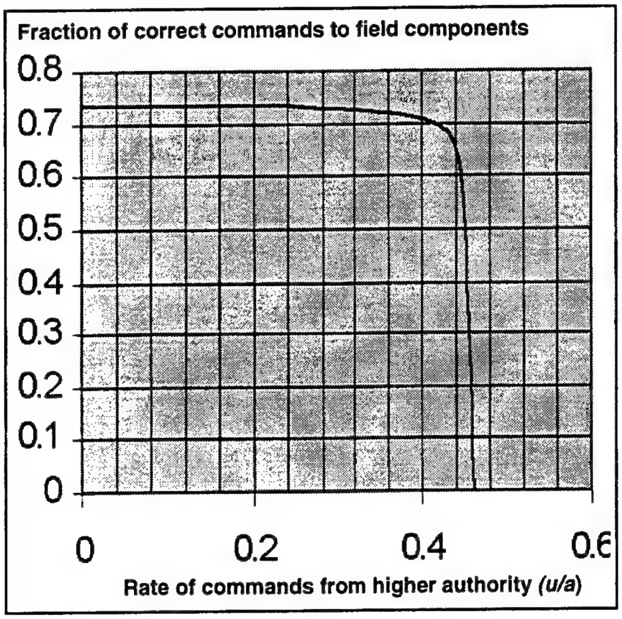


Fig. 9. As the rate of commands from higher-authority increases, the system reaches a collapse point.

The instability had a tendency to propagate through the hierarchical structure, i.e. overload at one location affects other locations by overloading the higher-level components which then in turn overload their subordinates. This is observed in the model by increasing individually any of the exogenous inputs from the field and then noting the rise of incorrect message rates in the entire architecture.

Figure 10 shows the stability region of the system as a function of the distribution of exogenous inputs to the two of the field components. Increase in either the combined operational tempo (i.e. the sum of the two inputs) or the difference in operational tempo across the lower-level units (i.e. the difference in the two inputs) leads to instability.

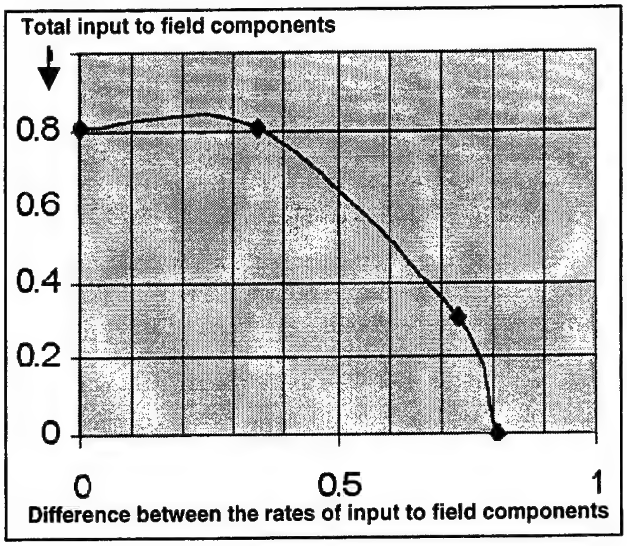


Fig. 10. The stability of the system is affected by both the overall rate of demands on the field components, and the differential across the components.

8. Approaches to Mitigating the ADO

Based on the results of the computational experiments we described, one can suggest several approaches to architecting and operating a C2 system in a way that reduces its susceptibility to the ADO.

Empowerment, mission-type orders. The onset of the ADO-related collapse can be delayed by reducing the number of requests sent from the Field to the C2 system (Figure 3). This suggests that ADO can be mitigated by enabling the Field entities to operate as independently as possible and to minimize the amount of cases in which they must call for the decision of the higher C2 system. Providing the Field units with maximum autonomy, use of mission-type orders and command by negation, contributes to the reduction of the K parameter and strongly improve stability of the overall system.

Prioritize and delegate. Avoidance of ADO can be achieved by dumping excessive messages – ignoring some of them and enabling the subordinate decision-makers to handle them autonomously. Perhaps this is how human C2 organizations in practice delay ADO – by learning to prioritize their decision-making load and to ignore the parts of it that appear less important.

Command-by-negation, not command-by-permission. The fact that dumping of decision load is often necessary to avoid ADO provides insight and support to the intuition that command-by-negation is advantageous as compared to command-by-permission. Let us remind the reader that in the command-by-permission protocol, a lower-level decision-making component detects a condition, formulates a plan for action, sends a request for permission to execute the action to the higher-level component, waits for the permission (or denial) to arrive, and then executes the action. In the command-by-negation protocol, the lower level component does *not* wait for permission, but proceeds to execute the action when the time is right, while being ready to abort the action if the higher-level component responds negatively. Clearly, avoidance of ADO by dumping at the higher-level component can be done more effectively in command-by-negation – if the higher-level component ignores the message, it does not prevent the lower-level component from executing the desired action. The same dumping of messages at the higher-level component in command-by-permission prevents the lower-level component from executing the necessary actions to exploit an opportunity or to block a threat. In both cases, dumping enables the C2 system to avoid ADO, but in the case of command-by-permission this avoidance leads to greater rigidity and passivity.

Minimize the need for coordination. Minimizing coordination loops, both vertical and horizontal, reduces susceptibility to ADO. Although reduction in coordination may appear counter-intuitive and controversial, one should note that human factors literature has been calling attention to the potential negative impact of coordination requirements for a long time . E.g., Morgan & Bowers [9] cite findings from Naylor & Briggs [10] as follows: "...the performance of operators in a simulated air-intercept task was superior when the subjects worked independently of one another. Decrements in performance were observed when operators were placed in an organizational structure that encouraged interaction among the operators." Experimental findings (e.g., Serfaty [11]) show that teams tend to perform better when they are able to communicate less under high-stress conditions. In a C2 team design, assigning tasks to minimize the need for coordination reduces the amount of knowledge the team members need to have about each other's roles, and the amount they need to communicate, resulting in better overall performance [12].

Insulate the weak link. Weaker decision-making components within the C2 system can accelerate the collapse of the entire system. It is advisable to insulate such a component from the rest of the system either by providing a greater degree of supervision or, if unavoidable, by allowing such a component to fail in its mission without expending excessive effort on the part of the superior component.

Diagnose on-line. In several experiments, we observed consistent symptoms of the onset of collapse manifesting themselves well in advance of the actual collapse. This observation suggests a possibility of introducing an on-line diagnostic mechanism to advise the C2 system that it must reduce its internal load.

9. Conclusions

We introduce the concept of Autocatalytic Decision Overload and argue that it may be a common phenomenon possibly responsible for major failures in command and control. The mechanism of ADO is rooted in positive feedback within the C2 system communication loops and in the decision workload-accuracy tradeoff. An inexpensive approach can be used to model and predict ADO phenomena under a variety of circumstances and for a variety of C2 architectures. Computational experiments suggest that susceptibility to ADO can be reduced by a number of means: dumping excessive load; empowering the lower echelons (mission-type orders); minimizing the need for coordination; using command-by-negation; insulating weak performers; applying on-line diagnostics.

Whether ADO is indeed one of the key mechanism responsible for catastrophic collapses of C2 systems in *real world situations*, remains a topic for future research. However, if a study of real-world cases confirms our hypothesis, then contributions of this paper — identification of the ADO phenomena and its failure mechanism, and the means to model, predict and mitigate the ADO — can be of significance.

Acknowledgments

This research was funded by DARPA's JFACC Program. A number of important suggestions were contributed by Jean MacMillan of Aptima, Inc.

References

[1] Kott, A. and Krogh, B. (1999), Toward a Catalog of Pathological Behaviors in a Complex Enterprise Control System, Proceedings of the DARPA-JFACC Symposium on Advances in Enterprise Control, San Diego, November 1999

[2] Van Creveld, M. (1985), Command in War, Harvard University Press, Cambridge, MA, pp.203-231

[3] Cohen, E. A., and Gooch, J. (1991), Military Misfortunes, the Anatomy of Failure in War, Vintage Books, NY, pp.214-228

[4] Burrough, B., and Helyar, J. (1990), Barbarians at the Gate: the Fall of RJR Nabisco, Harper and Row, NY, pp. 223-363

[5] Luce, R.D. (1986). Response times: Their role in inferring elementary mental organization. New York: Oxford University Press.

[6] Louvet, A-C., Casey, J. T., and Levis, A. H (1988), Experimental Investigation of the Bounded Rationality Constraint, in S. E. Johnson and A. H. Levis (ed.), Science of Command and Control: Coping with Uncertainty, AFCEA, Washington, D.C., pp. 73-82

[7] MathWorks Website: http://www.mathworks.com/

[8] SIMUL8 Corporation Website: http://www.simul8.com/

[9] Morgan, B.B. Jr., & Bowers, C.A. (1995). Teamwork stress: Implications for team decision making. In R.A. Guzzo, E. Salas, and Associates (Eds.) Team effectiveness and decision making in organizations. San Francisco: Jossey-Bass Publishers.

[10] Naylor, J.C., & Briggs, G.E. (1965). Team training effectiveness under various conditions. Journal of applied psychology, 49, 223-229.

[11] Serfaty, D., Entin, E. E., & Johnston, J. H. (1998). Team coordination training. In J. A. Cannon-Bowers & E. Salas (Eds.), Making decisions under stress: Implications for individual and team training (pp. 221-245). Washington, D. C.: American Psychological Association.

[12] Entin, E.E. (1999). Optimized Command and Control Architectures for Improved Process and Performance. Proceedings of the 1999 Command and Control Research and Technology Symposium, Newport, RI..

Dynamic Information Attributes in Distributed Collaborative Systems

Paul Labbé Zakaria Maamar Éric Dorion

Defence Research Establishment Valcartier (DREV)
Centre de recherches pour la défense, Valcartier (CRDV)
2459 Pie-XI Blvd. North, Val-Bélair (Québec), Canada, G3J 1X5

Paul.Labbe@drev.dnd.ca Zakaria.Maamar@drev.dnd.ca Eric.Dorion@drev.dnd.ca

Abstract

The appropriate exploitation of dynamic information attributes can improve the performance of man-in-the-loop systems in accomplishing certain collaborative tasks or missions. These dynamic information attributes control and allocate priority to information generation, processing and sharing. They depend at design time on user functions and systems and on the information exchange required for tasks. At operation time they depend dynamically on the structure of the assets at play, their spatial distribution and mobility ***(asset-mobility)****, and the information timeliness requirement. The agent-based architecture responsible for actuating the functions of interacting parties, from static to dynamic mobile agents* ***(soft-mobility)****, should display improved adaptivity and responsiveness in supporting a variety of operations where mobility is a must. These attributes and the agent-based architecture provide better information management capabilities, which previous studies have shown to provide substantial improvement in over-the-horizon targeting capability. We plan to use an extension of this overall approach in a technology demonstration.*

1. Introduction

Dynamic information attributes control and allocate priority to information generation, processing and sharing. The exploitation of these attributes by intermediate and end users can significantly impact the global performance of man-in-the-loop systems. These attributes can also be used in computing dynamic priorities or the Quality of Service (QoS) of processes and functions required in distributed collaborative systems. At design time they depend on user functions, on systems and on the information exchange required for tasks. At operation time they depend dynamically on the structure of assets (including people) at play: on its spatial distribution and mobility (asset/people-mobility), and the timeliness needed of decisions and actions if tasks and missions are to be accomplished successfully. Consequently, these attributes impact on the quality of shared information by collaborating entities, they provide the means for common knowledge and intent, and in the end they help to coordinate and synchronize the actions of an organization.

We summarize a six-layer architecture that allows the use of dynamic information attributes from design to operation time [1]. We propose an agent-based architecture to actuate the functions of interacting parties, from static to dynamic mobile agents (soft-mobility). Such an architecture should display improved adaptivity and responsiveness. We summarize the evolution of QoS from networks to soft entities, assuming that the agent-based architecture sits on an Object Request Broker (ORB) middleware and includes the NATO CSNI (Communications Systems Network Interoperability), which is typical of agent-based systems. Then we focus on information priority for which DREV model-based measures (MBMs) have shown a positive impact on over-the-horizon targeting (OTH-T) mission effectiveness. We introduce a utility function that computes information priority based on information value in an operational context for dynamically assigning agent-message QoS priorities. Finally, we present a possible hierarchical priority scheme.

2. Design and operational requirements

Joint operations (those among departments or services of a country) and coalition operations (those among cooperating countries) that exploit complex information technologies can be successful only if personnel and system assets are managed globally. Organizations are now beginning to exploit third-generation information systems to support operations that range from strategic to tactical, some with impact on long-term, global situations, others focusing on short-term, highly reactive situations in specific locations. Some of the objectives of these organizations to foster geopolitical movement towards desired states evolve at a daily pace, or more slowly. Others evolve more rapidly, and swift and forceful control actions are often required to attain desired end states or to correct situations that have degenerated. Systems to support such a variety of operations are necessarily highly complex, reflecting the purpose, authority and mobility of the assets needed. The design of supportive systems requires studying the nature of an organization, the end results to be effected, the functions needed and the information required. Success in attaining the desired ends depends on many factors, including:

1. a functional architecture that supports the organization,

2. a system architecture that offers the technical capabilities these functions require,
3. geographical distribution and sharing across the organization and over the operational area,
4. accuracy and timeliness of pertinent information,
5. training,
6. capability of developing a common intent, and
7. coordination and synchronization of actions.

It is convenient [1] to divide these issues into, first, those user and system functions whose data needs are independent of time and location (people, information, warehousing, systems and actuators) and, second, those in which both time and location attributes are essential. An effective architecture must address both requirements (as they apply at both design and operation times), and must be able to adapt to user needs that change after an asset has been deployed.

Basing an architecture on stated user requirements can be misleading: in large organizations it is often very difficult to extract precise user requirements (and little help is available to tackle this monumental work). Nevertheless, we must assume that at some point architects have acquired sufficient understanding of what an organization wants to accomplish and how it evolves for or adapts to future missions. Functions and processes that users exercise in attaining their organization goals are then identified. Relations among the users, functions and processes must be analyzed to identify how data is transformed into the information and knowledge required to pursue organizational goals. These relations also specify the nature, quantity, quality and accuracy of the data and information required to conduct specific organizational tasks. They constitute an important part of the cognitive dimension of an organization. These qualifications and requirements can be defined during the design of a functional architecture—more or less a static design—regardless of the location and temporal constraints of real operations. A dynamic or operational architecture takes the latter factors into account: the mobility of people and assets and the temporal, topological and geopolitical contexts.

Designing organization-wide systems as a single system from a global perspective offers several advantages. Such an approach increases the effectiveness of designers during the design cycle and allows more tractable global optimization during operations. Using the decomposition of the work domain—user functions and organizational goals—of an organization to define a support architecture buildsin interoperability, user support, measures of performance and effectiveness, traceability of information and transactions, fault tolerance and more. Vineberg's six-layer architecture of a global system follows [1]:

layer 6, user functions hierarchically structured with force-level functions spawning unit-level ones;

layer 5, system functions that support both static and dynamic distributed user functions;

layer 4, applications comprising software processes specific to missions;

layer 3, utilities combining software processes common to various tasks and missions;

layer 2, operating systems that allow diverse real-time assignments to processors; and

layer 1, resources that include physical components.

Adherence to this layered architecture offers the advantage of decoupling changes in threat from changes in technology: threat evolution affects the upper layers and technology the lower ones. Organization-wide systems (large, geographically distributed but with some mobility and having distinct security requirements) are extremely complex, expensive and difficult to track through attempts to upgrade while sustaining operations. Decoupling architecture changes due to threat evolution from changes due to advances in technology reduces risk considerably and allows architects to focus on organizational goals and user functions, while outside industry advances the technology that will eventually provide better support.

The thread concept introduced by Vineberg [1] identifies and records attributes (e.g., capacity, identification and appurtenance) of all elements required from layer 6 to layer 1 for a given user function, as an organization progresses with incremental system design and development. During operations, instantiating a user function and executing it means threading through the layers of the dynamic architecture imposed by the units in play, by their geographical distribution and by the role of each in the mission plan. The process identifies specific elements across layers that will be used if and when required. Some elements can be reused by or shared with other functions. Element sharing and dynamic configuration of the organization system at the time of an operation leads to the introduction of the concept of user-function priority to allocate resources and resolve deadlocks. This concept is linked to a utility function that computes information priority and QoS parameters that depend on the objectives of the user and of the organization (see section 5).

3. Responsive agent-based architecture

We present an environment in which multiple military participants could collaborate in spite of individual differences. The discussion supports user functions (layer 6 of Vineberg's user-centric architecture) by the required system functions (layer 5). To this end, a facility that supports the interoperability of the participating Command and Control Information Systems (CCISs) is needed. This facility relies on the concept of Software Agents (SAs), which can be collected into MultiAgent Systems (MASs). It is often claimed that communities of agents are much more powerful than any individual agent.

CCISs are evidently important for military land, naval, and air operations, and they are used increasingly in civilian applications such as air traffic control, search and rescue, and emergency services. In a military context, a commander makes decisions concerning his force deployment using the information supplied by the CCIS under his control and, possibly, by other friendly CCISs. For example, a commander is concerned with the positions not only of enemy units or targets but also of the friendly units of a coalition, so other CCISs may possess information that would improve the accuracy and completeness of his perception of the current battlefield. Ideally, the commander should be able to consult this set of CCISs without being aware of the structural and functional characteristics—the locations, languages, information semantics, etc.—of each. An interoperability environment is needed that will free military users or agency staff from worrying about the distributed, heterogeneous, and dynamic nature of the joint or coalition CCISs that hold information they need.

The motivation behind the design of an interoperability environment for CCISs is to provide an integrated view of all the aspects that are relevant to this environment. These aspects may include the CCIS structure, the roles played and responsibilities held by people in this environment, the flow of information within this environment and with the external world, the capabilities required by or available within this environment and the context in which it will be set up. Operational and staff requirements depend on extremely diverse, evolving geopolitical contexts, including war, peace-making/keeping, transitions from war to peace-making and from peace-keeping to war, other-than-war operations and disaster relief. MASs seem to be good candidates for a number of these aspects. For instance, an interoperability environment could be viewed as a collection of collaborative MASs, each corresponding to a CCIS and each containing several SAs of different types, with different roles and responsibilities.

3.1. What is a CCIS?

Information technologies are an inherent part of the commander's decision-making process. In particular, CCISs help commanders obtain an accurate view of the situation in which they are involved. A CCIS consists of a structure, tasks and functions [2]. The CCIS structure presents a set of facilities arranged to meet the objectives of the CCIS. CCIS functions are provided to initiate the tasks needed to meet these objectives. Tasks require structured facilities: personal, technical equipment, etc.

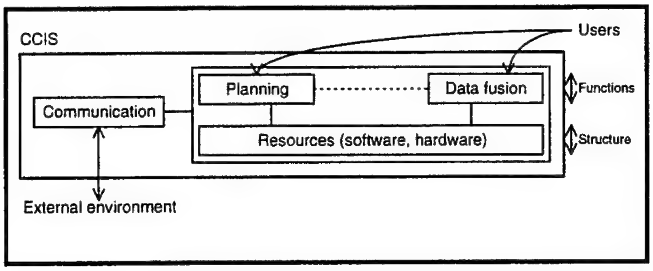


Figure 1 CCIS simplified architecture

Figure 1 presents a simplified architecture of a CCIS. The functions offered to military users range from planning and weather forecasting to logistics. They are built atop a support structure, both hardware (e.g., PC workstations) and software (e.g., a database management system). Certain functions of a CCIS receive formatted messages from other units or sensors through a communication module able to parse them. For example, the air-monitoring function receives messages from radar installations and from patrol planes, extracts their content and automatically updates appropriate databases.

3.2. What is a SA?

Researchers in Distributed Artificial Intelligence (DAI) have identified a broad range of issues related to the distribution and coordination of knowledge, and to actions in environments involving multiple agents [3]. These agents can be thought of collectively as forming a society. Agents can take different forms, depending on the nature of the environment in which they evolve. One particular type of agent, the SA, has recently attracted much attention. SAs are autonomous entities with the ability to assist users in performing tasks, to collaborate with each other to solve specific problems jointly, and to answer user queries.

Information technologies and communication capabilities evolve, and a single "mono-agent" approach cannot deal with the complexities of many separate agents (collaborative or competitive) evolving in the same environment and needing to interact in order to achieve a global goal, so agents are gathered into MASs. In such an environment, each agent's activities must consider the activities of the others, and research in MASs is concerned with understanding and modeling action and knowledge in a collaborative environment. The management of the distributed environment must coordinate behavior among agents and must detail how agents coordinate their knowledge, goals, skills and plans to make decisions for solving problems.

3.3. How to "agentify" a CCIS?

"Agentification" is the process of making a system behaves like an agent: to exhibit the main characteristics of an agent, such as autonomy and sociability, and to allow it to participate in a multi-agent environment. The approach proposed is to build an agent on top of a system.

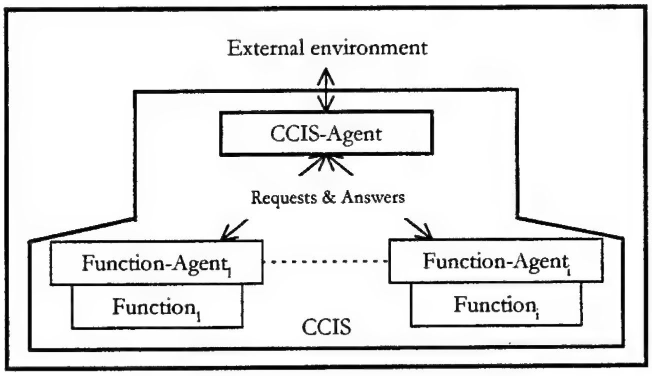


Figure 2 CCIS "agentification"

To "agentify" a CCIS we propose to introduce an agent called CCIS-Agent (Figure 2). A CCIS-Agent is the front-end of the CCIS-to-communication network; it acts on behalf of the CCIS and maintains its autonomy. It also advertises—through the services it provides—the different functions the CCIS performs. A typical service might be initiating the CCIS weather-forecast function.

As we have noted, a CCIS offers different types of functions to military users. Generally, these functions are very complex, for instance a planning function could be a distributed-object client/server application that runs on top of an ORB middleware. Because of this complexity and the fault-tolerance and efficiency criteria that a CCIS should meet, new types of SAs called Function-Agents must be introduced at the CCIS-Agent level, each corresponding to a specific CCIS function. The CCIS-Agent manages and monitors a group of Function-Agents (Figure 2). For instance, a request to the planning function of a CCIS is initially sent to the CCIS-Agent, which forwards it to the appropriate Function-Agent. MASs can be service providers or consumers—or both—running on top of an ORB middleware. The QoS of the middleware should be dealt with at functional and operational levels (the first at design time; the second at run time) to ensure that MASs can exchange services appropriately. For example, a requested service might be provided either remotely or locally, depending on the QoS specified in the middleware.

3.4. Proposed architecture

A variety of approaches to deal with the problem of interoperable systems can be found in the literature, among them Infosleuth [4], SIMS [5], and SIGAL [6]. All agree on the use of SAs as a means to develop such systems and all have elements in common, such as that all the SAs are static and cannot move to distant systems. Furthermore, all these approaches assume that the network infrastructure is fully reliable and has unlimited bandwidth (infinite channel capacity) for the transmission of information.

Based on these different approaches and on CCIS characteristics, we propose an architecture for the interoperability of CCISs (Figure 3). Several MASs form the backbone of this architecture. They interact about their respective CCISs by exchanging messages, either remotely or locally. In both cases, a facility called Advertisement Infrastructure is used, managed by an agent and containing a Bulletin Board and a Repository of Active-Agents.

We are aware that the Advertisement Infrastructure could be considered a bottleneck. In the mid-term, however, this potential drawback can be circumvented by duplicating the infrastructure and spreading it across networks. Excerpts or replicas of the centralized Advertisement Infrastructure would be distributed based on functional requirements at operation time. For highly reactive situations with limited channel capacity and poor reliability, such an extension ensures immediate access to priority information and provides global properties to local system functions and messages (priority and global properties are accessed locally). However, it means that a certain level of discrepancy must be accepted among replicas and the central repository. This immediate availability of priority is supported by the user-centric approach proposed. At operation time, systems and messages inherit priority ranges and properties the functional thread provided them in identifying resources needed for a user function.

In the proposed architecture, MASs consist of different types of SAs: Interface-Agents assisting users, CCIS-Agents invoking CCIS functions and satisfying user needs, Resolution-Agents, which also satisfy conflicting or contended user requirements or needs, Control-Agents managing MASs and, finally, a Supervisor-Agent managing the Advertisement Infrastructure. Interface-Agents, Control-Agents, and Supervisor-Agents are static, while CCIS-Agents and Resolution-Agents are soft-mobile. They can move across or to the Advertisement Infrastructure. Furthermore, the Resolution-Agent can move to other MASs. The various agents are:

1. **Interface-Agent**: assists users in formulating needs, maps needs into requests, forwards requests to the CCIS-Agent in order to be processed, and provides users with answers obtained from the CCIS-Agent.

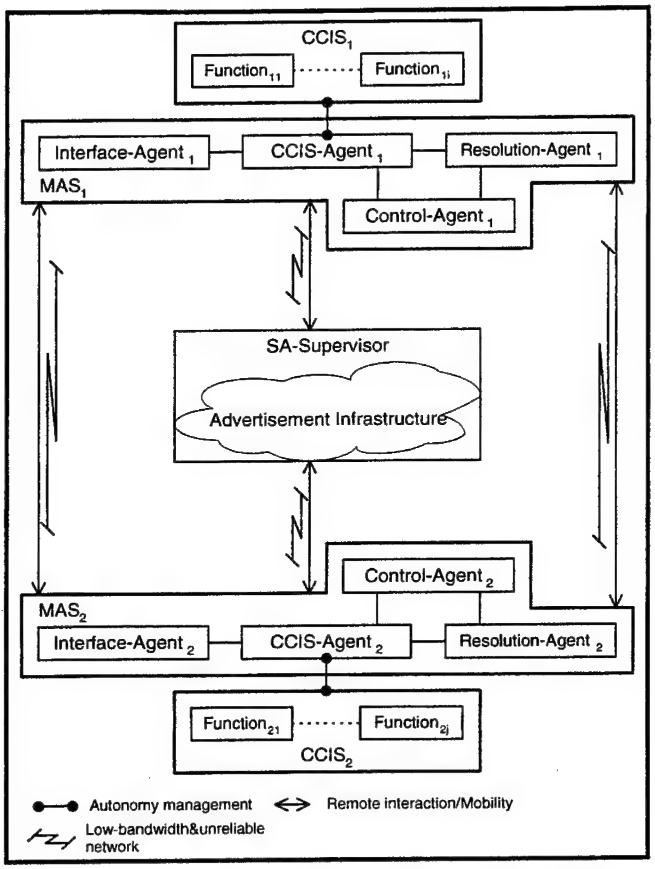


Figure 3 Architecture for interoperable CCISs

2. **CCIS-Agent**: processes user requests received from the Interface-Agent, but only if these requests require the involvement of the CCIS of this particular CCIS-Agent. In the proposed architecture, a CCIS-Agent has the ability to advertise its services by posting notes on the Bulletin Board of the Advertisement Infrastructure. To do so, the CCIS-Agent can either send a remote request to the Supervisor-Agent or can migrate to this infrastructure; the choice is based on the network status. In both cases, i.e., remote request or soft-mobility, a security level associated with the CCIS-Agent is used to identify the services this CCIS-Agent is authorized to advertise.
3. **Resolution-Agent**: processes user requests, but only if they are transmitted by the CCIS-Agent and can be met only with the involvement of several CCISs. In this situation, the resolution process requires that this Resolution-Agent collaborate with CCIS-Agents of other MASs. Thus, the Resolution-Agent moves to the Advertisement Infrastructure, consults the Bulletin Board, identifies appropriate CCIS-Agents through their offered services, goes back to its original MAS and, finally, designs the procedure needed to meet the request. This procedure is generally called a "route." The resolution process may require this Resolution-Agent either to interact remotely with the CCIS-Agents of the other MASs or to migrate to the MASs and meet their CCIS-Agents locally. Which action to take depends on network status and the number of CCISs required. As

with the CCIS-Agent, a security level is also associated with the Resolution-Agent. This security level is used to check Resolution-Agents entering the Advertisement-Infrastructure and the different MASs.

4. **Control-Agent**: in an environment consisting of soft-mobile agents, soft-mobility operations consist of shipping the agents through the net to other distant systems, authenticating them as they arrive, and installing them so they may resume their operations. In the proposed architecture, the Control-Agent of the MAS is in charge of all these steps. For instance, when the Resolution-Agent decides to move, it first interacts with the Control-Agent in order to be shipped to the desired MAS. Furthermore, Control-Agents maintain the coherence of their MASs by keeping track of the Resolution-Agents entering and leaving these MASs.
5. **Supervisor-Agent**: manages the Advertisement Infrastructure by receiving CCIS-Agent advertisements, sets up a security policy to monitor the CCIS-Agents and Resolution-Agents accessing this infrastructure and, finally, installs CCIS-Agents and Resolution-Agents so they can resume their operations in this infrastructure. In our architecture, the Supervisor-Agent uses the Repository of Active-Agents to register all the CCIS-Agents and Resolution-Agents that are authorized to enter and leave the Advertisement Infrastructure.
6. **Advertisement Infrastructure**: in an interoperating environment, CCISs are generally spread across networks and rely on low capacity and unreliable channels for communication. Moreover, a military user may use his Combat Net Radio to send and request information or may rely on mobile devices, such as portable computers, that are only intermittently connected to networks. In the proposed architecture, to avoid overloading the network, CCIS-Agents and Resolution-Agents migrate to the Advertisement Infrastructure in which CCIS-Agents advertise their services by posting notes on the Bulletin Board, whereas Resolution-Agents consult the Bulletin Board to identify the CCISs that are required to satisfy user needs.

4. Evolution of QoS

In a mobile environment where resources are scarce and channel capacity is intrinsically low, strategies and tools have to be developed to ensure the efficiency of distributed applications. QoS is a concept that encompasses the tools, strategies, methodologies and characteristics that ensure the performance and efficiency of a system. Of course, QoS management would not be needed if computing and channel capacity were infinite, so one might think that the need for QoS management would decrease as network capabilities increase. In fact, however, as network capabilities such as channel capacity increase, requirements appear for new functionalities—along with the computer capacities needed to meet them—canceling out much of what has been gained. Therefore, QoS remains a major concern [7, 8].

4.1. Typical network QoS attributes

Good examples of QoS attributes can be drawn from IP (the network layer of the Internet protocol) and CLNP (connectionless network protocol). These networks use type-of-service (ToS) fields to indicate something special about packets that a source would like to see accommodated and that routers can intelligently supply [9]. A one-byte IP field includes: 1– precedence (priority), usually 0 for lowest and 7 for highest; 2– normal delay or low delay; 3– normal throughput or high throughput; and 4– normal reliability or high reliability. With these definitions, the QoS of "normal" IP is low priority, high delay, low throughput and low reliability, which led Perlman to ask, jokingly: "Would you buy a network from someone who defined normal that way?" [9].

For CLNP, QoS is divided in two: "quality of service management" (QoSM) and priority. QoSM gives the receiving entity the ability to indicate congestion directly to the source, while IP relies on routers to send a specific packet to the source when congestion occurs. In both cases, this momentarily increases traffic in congested network segments. Error reports and other network feedback allow some management and reporting of inconsistencies between source QoS requests and that available from the network or the destination during operation.

4.2. Cooperative software entities

The same principles that apply to packet delivery over IP networks apply to the delivery of the object messages required by distributed software architectures such as CORBA. QoS for such distributed applications depends on:

1. minimizing message delivery delay,
2. minimizing jitter between similar message calls,
3. stabilizing throughput of message calls and the responses of objects, and
4. managing information based on message priority.

The end-to-end principle and the need for negotiation between objects (cooperative software entities or agents) to achieve an agreement on operating level still hold. Since the middleware is composed of objects, the end-to-end principle implies that a negotiation must take place between objects in the application (client and server) and those in the middleware. Therefore, built-in mechanisms are needed in the middleware to achieve QoS-awareness in distributed systems. Also, the middleware should be responsible for initiating QoS negotiations with the next lower communications level, and so on. The CORBA Messaging specification [10] includes provisions for QoS.

4.3. New approaches to QoS

BBN Systems [11, 12] has proposed an interesting out-of-CORBA QoS solution: the QuO (Quality of Objects) Toolkit. The concept revolves around the powerful metaphor of contracts between objects. A QuO contract is an entity that states one or several regions of quality in which an object can operate. An agreement between objects takes place when one or more of their two regions of quality overlap. If QoS changes over time (which is common), other regions of quality are entered and objects are informed. The objects can either cease to operate or can adapt. An advantage of this approach is that it is dynamic. A distributed application can adapt to its QoS environment provided that there is code to be executed when entering another quality region. The drawback is that the approach is out-of-CORBA and thus less portable, less maintainable and less scalable.

The notion of end-user QoS is not unique to military systems. Among issues that impact the effectiveness of an organization most strongly is the QoSs that end users observe [13]. Two major components affect the perceived QoS (transposed from educational to military systems): quality of content (information and actuation capabilities) and quality of plan (strategies matching desired geopolitical changes). Network QoS and end-user QoS serve very different purposes, but the former impacts the later, not the reverse. End-user QoS issues are better served by QuO.

4.4. Typical soft-entity QoS attributes

In an international experiment in distributed simulation [14], a requirement was noted for resolving the discrepancies between the QuO and QoS required by an application and the end-to-end QoS available from the current state of network facilities along the route. Components of the distributed simulator were installed at two locations, one in Australia and the other in Canada. Local routers and intrinsic satellite-link characteristics for this large exercise scenario imposed a minimum one-way delay of about 175 ms, while real-time requirements for tightly coupled simulated entities were under 100 ms end-to-end (up to 300 ms for loosely coupled entities). Consequently, fast entities were not fully synchronized, and their dynamic behavior had to be smoothed using first-order dead reckoning. In this experiment, delay was the critical QoS factor; link capacity was not a problem except that as the number of simulated vehicles increased, the percentage of packet loss increased (e.g., 1.6 % for 18 and 20 % for 42 vehicles over 56-kb/s ISDN). Packet losses and delay satisfied the simulation QoS requirements for loosely coupled entities.

Network latencies forced a minimum delay that needed to be addressed by changing either the network design or the maximum acceptable delay in the simulation. The latter approach was chosen because it could be more easily controlled. However, improved QoS delay capabilities in the network would have allowed requests for 100-ms and 300-ms QoS to be met using different network services. The 100-ms traffic would have received a higher priority or precedence than the 300-ms traffic. Other attributes would allow the global quality of the distributed simulation to be dealt with by combining delay, reliability and data volume requirements evaluated by a real-time utility function accounting for current network delay/latency, reliability and current effective user-channel capacity. Attaining such QoS resolutions would have improved the fidelity and synchronization of the distributed simulation and allowed larger scenarios and a higher pace for entities such as aircraft. Communications costs could have been optimal, i.e., minimum for the required services used.

4.5. Asset-mobility imposing adaptivity

Specific bandwidth-routing protocols to support QoS in wireless networks have been proposed and studied for functions that require real-time communications. The QoS routing protocol proposed in [15] computes bandwidth information in a multihop-mobile network. It considers only bandwidth as a QoS, omitting error rates on the assumption that a bandwidth guarantee is one of the most critical requirements for real-time applications. Results obtained with this QoS routing protocol showed adaptation to the decrease of the bandwidth (effective channel capacity in [16]) as the relative velocity of mobiles increases. Alternatively, QoS routing when end-to-end delay requirements are considered can be difficult to compute, although some heuristics show promising results [17]. In reality what is needed is: 1– sharing knowledge of the currently achievable QoS [18] with appropriate error-control techniques, and 2– negotiating between agents to use network services with a variable-bit rate or an available-bit rate to achieve a given error rate (maximum bit rate for a prescribed error rate). Most real-time applications more readily accept the loss of few data updates than an increase in average delay. In CCIS, volatile tactical data about the current position of an aircraft must be updated frequently: the latest positional information has more impact on mission effectiveness than the retransmission of any previously lost updates. Corrective measures that reduce the error rate of updates below a required level increase information exchange latency and delay. However, for this same aircraft, information on allegiance requires a reliable multicast service and acknowledgments from relevant addresses.

4.6. Smaller footprint requirement

Middleware technologies are often reproached for their memory requirements: the memory footprint, largely determined by the static and dynamic size of the object request broker's core, object adapter and the stubs/skeletons generated by their compiler [19]. The same reference shows that interpreted and optimally compiled versions can generate footprints acceptable for handheld or palm computers. Thus such middleware is suitable for most asset- and soft-mobile multimedia applications, and certainly for CCISs.

4.7. QoS of the NATO CSNI Project

Agreements (1986–96) and work among Canada, France, Germany, the Netherlands, the United Kingdom and the United States for conducting the CSNI project [20] have led to design concepts and demonstration results to support stationary and mobile entities distributed worldwide. Message-data ranged from real-time, high-volume such as image and voice at nominal reliability—the large traffic—to less real-time-demanding but still time-critical, with needs for high reliability and confirmation of reception by addressees—the small traffic with the highest importance.

Military operations still rely on voice messages, requiring C2 systems that support non-secure and secure real-time voice services as well as data and multimedia. Legacy systems include dedicated special-purpose voice communication components. New-millennium military systems and network designs strive to decrease the illogical proliferation of special-purpose systems (stovepipe crisis or anarchy) by moving applications onto open systems and general-purpose, connectionless, router-based networks. Moving applications such as real-time voice, which requires a guaranteed QoS, or bit-pipe, over to general-purpose "best-effort" connectionless networks requires special attention since typical connectionless networks cannot guarantee either timely delivery or a maximum delay, and may actually drop packets when they are congested.

The technique explored in CSNI to address the QoS guarantee requirement for real-time applications was the reservation of network resources for traffic from a source to one or several addressees. More specifically, CSNI-2 investigated the implementation and performance issues of the Reservation Protocol (RSVP) to reserve network resources such as bandwidth and maximum delay for a real-time voice application (VAT).

Dynamic allocation and sharing of network resources for distributed real-time applications implies granting sufficient network resources to meet the required QoS. Legacy networks were not designed to guarantee a maximum end-to-end delay using only the "best effort" point-to-point transport services offered by existing connectionless networks. Recent communications networks such as CSNI have implemented multicasting, real-time services and service guarantees. The RSVP tested in CSNI offers these capabilities by reserving bandwidth on end systems and routers (or intermediate destinations) along the path from source to destination(s) across an IP-based router network. For interacting users such as voice that need real-time capability, this new network service matches or improves on the service provided by dedicated circuit-switched communications systems, at a much lower overall cost. It allows available network assets to be used dynamically, varying over time and the location of user needs. A generic resource reservation system is used across the network to negotiate QoS and to allocate resources. This reservation can be done via the Advertisement Infrastructure of the SA-Supervisor of the agent-based architecture proposed in this paper.

Several QoS attributes and management capabilities were identified as necessary for worldwide multimedia interoperability among stationary and mobile entities of NATO. Service guarantees depend on QoS management allowing [20]:

1. designers/users to specify transmission characteristics for the application,
2. translation of the QoS parameters between different layers when their specifications are different,
3. negotiation of the QoS parameters at the time connections are established, and possible renegotiation as required (especially for mobiles in a hostile environment),
4. mapping of QoS parameters to resource requirements,
5. admission control, reservation and allocation along the path between the sender(s) and receiver(s), and
6. QoS monitoring to ensure that the specified threshold values are not exceeded.

Resource reservation in end systems and routers is the feature needed to guarantee QoS to users. Otherwise transmission over unreserved resources would lead to dropped or delayed packets, in violation of user requirements. Constraint fulfillment required for maintaining QoS guarantees over the duration of service includes [20]:

1. time constraints, e.g., delays and delay variations,
2. space constraints, e.g., buffers,
3. device constraints,
4. capacity constraints, e.g., channel and system capacity for data transmission, and
5. reliability constraints, e.g., forward-error and feedback-error control techniques.

QoS parameters for a service negotiated at connection time need to be guaranteed and registered for the duration of a service, and renegotiation may be needed in a dynamic environment. Bounds for parameters such as delay, loss and jitter must be maintained for the duration of the service connection. Conventional transport protocols such as TCP (transmission control protocol) or TP-4, which were designed to provide reliability by employing end-to-end acknowledgment and retransmission without providing the concept of time-constrained services, cannot meet these requirements. Required protocols address such issues as [20]:

1. resource reservation protocols to establish connections that satisfy QoS requirements, and
2. resource administration functions such as admission and monitoring multicasting functions, and lightweight transport protocols.

It is worth noting the parallel between our agent-based architecture and CSNI [20]:

> The information on a CSNI agent is a **Management Information Base** (MIB), which logically encompasses configuration and status values normally available on the agent system. A specific type or class of management information is called a MIB object (for example, a system description or an interface status). The existence of a particular value for a MIB object in the agent database is called an **instance**. Some MIB objects have only a single instance for a given agent system (for example, system description). Other MIB objects have multiple instances for a given agent system (for example, interface status for each interface on the system).
>
> The MIB objects are defined using the Internet-standard Structure of Management Information (SMI) and compose a virtual data store on the agent system. This structure is defined by *RFC 1155: Structure and Identification of Management Information for TCP/IP -based Internets* and amended by *RFC 1212: Concise MIB Definitions*. Together, RFCs 1155 and 1212 define the structure of management information for Simple Network Management Protocol (SNMP) based management. SNMP agents contain the "intelligence" required to access MIB values.

5. Impact of information management (IM) learned from legacy systems

Successful coalition and joint operations depend on the completeness, pertinence, accuracy and timeliness of the information shared by participating units for planning and deciding course of actions and for controlling effectors to reach mission objectives. Communication delays are one of many factors that degrade the quality of this shared picture. A priority assignment scheme has been presented in [21] to reallocate the available communications capacity based on the value to the end user of the information contained in individual messages, and on the contribution of each item to overall mission success. Analysis of data from military exercises suggests that even a simple such modification to communications procedures could increase the effectiveness of one critical command-and-control task, over-the-horizon targeting (OTH-T), by as much as 22%.

An examination of the Global Command and Control System (GCCS) transmission-queue management abilities (outgoing messages) for broadcast in the Force Over-the-horizon Track Coordinator (FOTC) function and other data-forwarding functions reveals a variety of IM possibilities. Outgoing messages may be selected for given geographical areas that match the Areas of Operational Interest (AOIs) of deployed units and different transmitting strategies may be defined, each with its own out-going message queue. Based on live exercise data collected, it is possible to use the time currency of the reports to select data older or younger than a given criterion, allowing the implementation of first-in/first-out (FIFO) or last-in/first-out (LIFO) strategies. Similar strategies were studied and tested with agent-based technologies [22]: push, pull and sentinel-style monitoring. Either one or a combination of these strategies was found to be needed in most military applications addressed.

Schemes or information-management heuristics (IMHs) that prioritize data to be sent based on its information value to a task or a mission cannot currently be implemented easily within the GCCS software. Wrapping current GCCS in an agent within the proposed agent-based architecture would certainly ease the testing and implementation of schemes that exploit dynamic information attributes needed to optimize distributed collaborative systems globally. Consequently, we expect that exploiting information attributes in an agent-based CCIS architecture would improve OTH-T effectiveness by at least the amount reported in [21], since this measure supports the desired information management strategies. Based on our experience, we expect the proposed architecture and approach to impact other missions similarly. Such a potential improvement helps to justify plans for improving information management and cooperative engagement capabilities, as described in a recent naval requirements document [23], which encompasses an emerging unified joint and coalition philosophy from Canada and its allies that builds on the best practices in the field.

5.1. Priority based on value to missions

In Coalition operations, a large variety of information must be exchanged with widely differing needs for QoS. To meet these different needs, certain radio-communications assets can exploit priority schemes, as demonstrated in the NATO CSNI project [20]. The value and timeliness of the data to be exchanged can be assessed by the information node, in accord with requirements for the successful accomplishment of each addressee's tasks. A CSNI-like node can provide information on its current QoS. All of these items of information can be used to improve the quality of the shared information.

Assigning priority to messages, packets or cells in terms of task or mission effectiveness requires the extraction of the information each contains; that is, to find what each piece of data means to its end user. Then from knowledge about the missions and tasks to be accomplished and from established time and location attributes for information for each task, the value of the data and related timeline requirements can be assessed. Through an appropriate utility function, the current priority of the data is computed from the time-dependent value of contained information. The value of certain data may depend on information in other data to be sent simultaneously; the value is thus conditional upon the ability to send the combined data within a given time interval. All time-dependent pieces of information stacked in such priority queues must be reestablished prior to each transmission opportunity.

In such scenarios and architectures, the communications nodes provide extensive QoS information such as:

1. current and previous network status,
2. expected time to the next transmission opportunity,
3. maximum amount of data that can be transmitted in one transmission opportunity,
4. estimated minimum time required to transmit a particular amount of data to another node on the network,
5. estimated probability of error-free delivery to each addressee, and
6. estimated delivery delay of correct data to each addressee.

It is assumed that functional nodes (units of the user-centric architecture) include all that is needed to assign priority to information based on task and mission knowledge, and that communication nodes know how to route data with a given priority and QoS to a list of addressees. Also, we assume complicity between the network node and the system node in managing the information transmission queue at each unit. Once information enters the communications network, it becomes data to be transported and its meaning to a user is irrelevant to the network components, except for attributes such as destination(s), QoS and priority.

5.2. Computing context-sensitive priority

In evaluating the value of information or of a function to be executed next, one has to consider the context in which they will be used. For a user responsible for OTH-T, information changes (and related functions and processes) in hostile tracks within its AOI are critical, while changes in other tracks or in those outside its AOI may have less immediate impact on mission effectiveness.

The proposed assessment of the value of information for a task is based on the following parameters:

1. Importance, I: the significance of a context relative to all others.
2. Potential, P: the relevance of information in context.
3. Quality, Q: the goodness of information, e.g., accuracy.
4. Currency, C: the freshness of information.

Assuming constant I, P, Q and C parameters based on the message information content or attributes and a particular context, the generic utility function proposed for assigning priority to information in an operational context is (adapted from [24]):

$$\mathrm{Priority}(i, \alpha) = w\, \mathrm{I}_{\alpha} \cdot (w_{\mathrm{P}}\, \mathrm{P}_{i\alpha} + w_{\mathrm{Q}}\, \mathrm{Q}_{i\alpha} + w_{\mathrm{C}}\, \mathrm{C}_{i\alpha} + \mathrm{X}) \quad (1)$$

where:

w = priority weight, w_{P} = potential weight,
w_{Q} = quality weight, w_{C} = currency weight,
I_{α} = importance of context α,
$\mathrm{P}_{i\alpha}$ = potential of information item i in context α,
$\mathrm{Q}_{i\alpha}$ = quality of information item i in context α,
$\mathrm{C}_{i\alpha}$ = information item i currency in context α, and
X = additional factors yet to be determined that can include dynamic properties of the parameters, including time dependence.

Reference [24] provides a description of the various parameters that is sufficient to prepare test scenarios and scheme implementations. Hierarchical relations among information items in context are also presented.

5.3. Hierarchical priority

In a hierarchical organization for theater battle, a theater-battle commander (TBC) receives the highest range of priority [1, 25]. The TBC delegates a subset of this range to task force commanders (TFCs). Unit commanders (UCs) of a task force receive their respective sub-subsets of priority ranges from their TFC. When a TFC moves into a theater it receives a new priority range from the concerned TBC. When a UC moves from one TFC responsibility to another, the previous range of priority is relinquished and the new one adopted.

This dynamic aspect of priority ranges of users and functions at operation time must be combined in our utility function to follow the imposed hierarchical order. We can associate the notion of importance I_α and its priority weight *w* to this hierarchical order with other operational factors. For example if *w* ranges for TBC, TFC and UC are (0, 10), (0, 9) and (0, 8) respectively, assuming other parameters of the utility function being the same, then higher authority will receive higher priority. However, if a TBC wants to use resources for a routine videoconference, even though his priority range is the highest the asset would still respond to a fast air threat, since the priority utility function will use parameter values accepted by the user community to ensure such action. The utility function will deliver a higher priority according to importance, context, potential, timeliness and other factors deemed necessary by users, factors defined at design time and computed in context at operation time using spawned priority inheritance actuated by the thread.

Atomicity of information and function is important in priority computation and assignment. Lumped models may lead to poor optimization and inappropriate control and use of assets. Currently, track data combines positional, allegiance and other attributes. As indicated previously, positional updates must suffer minimum delay, while allegiance and order require high reliability and confirmation of delivery to action addressees. Combining these requirements imposes expensive and probably impractical infrastructures, because efficiency opposes redundancy, according to information theory. That is, reliable sharing of information imposes a delay because of the protocols and control mechanisms required. Consequently, to optimize the sharing of needed timely and reliable information, track updates should be done incrementally for the track field that needs it (see Handbook Five recommendations).

Each track attribute should be assessed as an item of information. At design time this item is associated with user functions. During operations, for a given task and context it receives precedence and QuO or QoS (via the actuated architecture thread). These could be computed by a priority utility function as defined here. Priority, QuO or QoS can be computed to match the dominant requirement for each item of information, ranging from the fastest aging to the highest reliability. Then incremental updates of position of a hostile track will receive high priority from an OTH-T coordinator for low delay (the shortest if it is a high-threat, fast air track) with less stringent requirement for reliability (e.g., bound by a maximum acceptable error rate and acknowledgement after *n* incremental updates). These updates use a unique sequential number associated with the track. Positional update reporting traffic should be almost continuous over time, with a medium net average traffic load (much less than video).

Similarly, allegiance reporting will use high reliability sharing mechanisms that offer confirmations from addressees. Determination of the allegiance of own units is facilitated by their own reporting. Determining the allegiances of other entities requires the exploitation of various sources of information—sensors, intelligence and databases—and usually involves staff deliberation. Allegiance updates are rare compared to positional updates, occurring only when a user or system observes such a change and wants to confirm it, or requires confirmation that all or specific allied units are aware of a change. Traffic generated by allegiance updates is low even though they use high-load, reliable sharing mechanisms. Allegiance reporting should show time-scattered traffic bursts with an overall average traffic load that is negligible.

A function or information priority scheme impacts networks, assets, systems and users in a given command thread. It is an essential step towards battle resource optimal utilization and user-centric responsiveness. It provides enabling techniques and support to force coordination and synchronization.

5.4. Global QuO priority management

QuO, QoS and priority assignment need to be negotiated in a virtual centralized facility. In the proposed agent-based architecture this negotiator can be part of the Advertisement Infrastructure at the SA-Supervisor. The advantage of centralized management is the global view of the organization, a unified perception of theater asset state. This allows organization-wide computation of QuO, QoS and priority. In dynamic organizations with uncertain connectivity and non-ideal communications, part of this centralized knowledge should be distributed to improve function responsiveness and robustness.

The need for a global view is supported by allied studies. However, there is a requirement to maintain a distributed repository of a part of the global knowledge for the purpose of managing assets when highly reactive responses are needed. This can be exemplified as follows:

1. We are under attack... we do not have time to verify our ship readiness... ask for latest rules of engagement from TBC... last update received should do!
2. End of war in a theater [TBC], blue-on-blue engagement preemption by TFC, preempt engagement initiated by own unit [UC].
3. Start of war in a theater [TBC] and plan force deployment [TBC+TFC], deploy force asset and plan unit tasks [TFC], unit initiates assigned task and plans actions [UC].

6. Conclusions and Recommendations

The impact on mission effectiveness of adopting dynamic information attributes and exploiting agent-based architectures for CCISs cannot be measured as directly as it can be for most weapon systems. Further studies are required to explore the approaches reported. Assuming that the proposed approaches implement efficiently the concepts reported in [21], we expect that they may improve OTH-T effectiveness for hostile surface contacts by 22%. More complicity between information sources, communication networks, systems used by staff and software agents representing the information requirements of the end users may prove to be particularly cost-effective in the long run [22]. We will recommend using a consolidated version of the proposed agent-based architecture and attributes in a Canadian Technology Demonstration.

This paper builds on [26], which provides AUS-CAN-NZ-UK-US (Australia, Canada, New Zealand, United Kingdom and United States) endorsed guidelines for the procurement of national communications, command, control and intelligence systems for the compilation and sharing of accurate information used by commanders.

7. References

[1] Vineberg, M., *A Proposed User-centric Battle Management System*, in Proc. of the 1999 12th Ship Control Systems Symposium, The Hague, The Netherlands, 16 pages, October 1999.

[2] Malerud, S., Feet, E.H., and Thorsen, U., *A Method for Analysing Command and Control Systems*, in Proc. of the 1998 Command and Control Research and Technology Symposium, Naval Postgraduate School, Monterey, CA, 29 June to 1 July 1998.

[3] Jennings, N., Sycara, K., and Wooldridge, M., *A Roadmap of Agent Research and Development*, Autonomous Agents and Multi-Agent systems, Vol. 1, No. 1, pp. 7-38, 1998.

[4] Bayardo, R., Bohrer, W., Brice, R. et al., *Infosleuth: Semantic Integration of Information in Open and Dynamic Environments*, in Proc. of the 1997 ACM International Conference on the Management of Data (SIGMOD), Tucson, AZ, 1997.

[5] Knoblock, C.A., Arens, Y., and Hsu, C.N., *Cooperating Agents for Information Retrieval*, in Proc. of the Second International Conference on Cooperative Information Systems, Toronto, Canada, 1994.

[6] Maamar, Z., Moulin, B., and Bédard, Y., *Software Agent-Oriented Frameworks for the Interoperability of Georeferenced Digital Libraries on the World Wide Web*, The SIGAL Project, Kluwer Academic, Boston, 1999.

[7] Bragg, A.W., *Quality of Service: Old Idea, New Options*, IT Professional: Technology solutions for the enterprise, Vol. 1, No. 5, pp. 37-44, 1999.

[8] ISO/IEC, *Information Technology - Quality of Service: Framework*, ISO/IEC 13236, ISO/IEC, Geneva, December 1998.

[9] Perlman, R., *Interconnections: Bridges and Routers*, Addison-Wesley, Reading, MA, 1992.

[10] Object Management Group (OMG), "Common Object Request Broker Architecture (CORBA)," 1999.

[11] Zinky, J.A., Bakken, D.E., and Schantz, R.E., "Architectural support for quality of service for {CORBA} objects," in *Theory and Practice of Object Systems*, vol. 3: John Wiley & Sons, 1997.

[12] Zinky, J.A., Bakken, D.E., and Schantz, R.E., *Overview of Quality of Service for Distributed Objects*, in Proc. of the Fifth IEEE Dual Use Conference, Utica, New York, May 1995.

[13] Vouk, M.A., Bitzer, D.L., and Klevans, R.L., *Workflow and End-user Quality of Service Issues in Web-based Education*, IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 4, pp. 673-687, 1999.

[14] TP2 JSA TP2, *A User Guide on Setting up ModSAF International Links*, Communications Research Centre, Ottawa, June 1999.

[15] Lin, C.R. and Liu, J.-S., *QoS Routing in Ad Hoc Wireless Networks*, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, pp. 1426-1438, 1999.

[16] Labbé, P., *Mobile Networks: the Effect of Relative Node Velocity on Effective Channel Capacity*, in Proc. of the 1999 International Conference on Telecommunications (ICT '99), Cheju, Korea, Vol. 1, pp. 147-156, 15-18 June 1999.

[17] Guérin, R.A. and Orda, A., *QoS Routing in Networks with Inaccurate Information: Theory and Algorithms*, IEEE/ACM Transactions on Networking, Vol. 7, No. 3, pp. 350-364, 1999.

[18] Capone, J.M. and Stavrakakis, I., *Achievable QoS in an Interference/Resource Limited Shared Wireless Networks*, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 11, pp. 2041-2051, 1999.

[19] Gokhale, A.S. and Schmidt, D.C., *Optimizing a CORBA Internet Inter-ORB Protocol (IIOP) Engine for Minimal Footprint Embedded Multimedia Systems*, IEEE Journal on Selected Areas in Communications, Vol. 17, No. 9, pp. 1673-1706, 1999.

[20] NATO, *CSNI-2 Final Report*, CSNI/2/D/003/Rev1, NATO, December 1996.

[21] Labbé, P., *Information-management Heuristics for Improved Coalition-operations Effectiveness*, in Proc. of the 1999 IEEE Military Communications Conference (MILCOM '99); Into the Next Millennium-Evolution of Data Into Knowledge, Fort Monmouth & Atlantic City, NJ, 31 October - 3 November 1999.

[22] McGrath, S., Chacón, D., and Whitebread, K., *Intelligent Mobile Agents in Military Command and Control*, in Proc. of the Agents in Industry / Autonomous Agents' 2000, Barcelona, Spain, June 2000.

[23] Canadian Navy, *The Naval C4I Way Ahead (Draft)*, Department of National Defence Canada (Cdr S. Bertrand), Ottawa, May 1999.

[24] Flores, A., Blogett, D., and Auns, V., *Canadianization of Handbook 5, CSIS Task 109*, LMC 500000109 Rev. 1, Lockheed Martin Canada Inc., Montréal, Canada, January 1999.

[25] Vineberg, M., *A Proposed Navy Battle Management Architecture*, in Proc. of the 1991 Symposium on Command and Control Research, Washington, D.C., pp. 120-133, December 1991.

[26] AUS-CAN-NZ-UK-US Organization's MONIME Ad-hoc Working Group (Co-author), *Handbook 5: Guidelines for Maritime Information Management*, The Management of Organic and Non-organic Information in the Maritime Environment (MONIME) Ad-hoc Working Group (Command, Control and Communications Committee), 19 April 1997, UNCLASSIFIED.

Dynamic Control of Acoustic Communication Routes for an Autonomous Undersea Distributed Field of Sensors

Mark D. Hatch
SPAWAR Systems Center Code D722D
hatch@spawar.navy.mil

Abstract

The Deployable Autonomous Distributed System (DADS) Intra-Field Data Fusion Project is developing technology to fuse sensor information from a field of autonomous sensor nodes and to dynamically control the field. The field consists of three different types of nodes in littoral waters, which operate on batteries and communicate underwater via acoustic modems. Sensor nodes contain acoustic sensors, electric field sensors, and vector magnetometers. These nodes collect and process data, fuse the acoustic and electromagnetic data available within the node, and forward contact information to a master node. The master node fuses the sensor outputs and also dynamically controls the power usage in the nodes to maximize system lifetime. Data are sent to an command center via gateway nodes using RF communications. This paper will concentrate on the network control methodologies being developed for the master node.

1 Introduction

The Deployable Autonomous Distributed System (DADS) Intra-Field Data Fusion Project seeks to develop technology to support a field of autonomous sensors in shallow water [1]. Technologies under development include the fusion of data within the field and control of the communications network and other functional processes to extend the life of the field. This project, sponsored by Dr. D. H. Johnson at the Office of Naval Research, is an integral part of a broader thrust which is addressing the other technologies required for the implementation of the overall DADS concept. The concept utilizes three different types of nodes, which make up a network. *Sensor nodes* are small nodes that sit on the ocean floor and contain acoustic sensors, electric field sensors, and vector magnetometers. Data are collected from the sensors, processed, and locally fused in the node. The node then forwards contact information to a *master node*, which controls the field and fuses the data it receives from the various sensor nodes. Master nodes send their data acoustically to *gateway nodes*, which communicate with a command center via RF communications. Each of the nodes will run on battery power and communicate with each other via underwater acoustic modems.

The communication range of each sensor is expected to be limited by the poor acoustic propagation conditions that exist in shallow water. The design of the DADS field calls for several tens of sensor nodes and very few master nodes. This architecture creates the requirement for each message generated by a sensor to be relayed between several sensor nodes until it reaches a master node, and for command messages from the master node to the sensor nodes to be relayed in a similar manner. The messages that are created and relayed in the field are expected to consume a great deal of the battery power.

There are many unique problems and opportunities for research and technology development for such a sensor field. Among them is the limitation on energy based on using batteries to power the nodes. This paper will describe work performed by Wagner Associates and SPAWAR Systems Center San Diego on the dynamic control of the field in order to maximize system lifetime [2].

2 Field Initialization and Communications

When a DADS field is initially laid down, there is an initialization procedure that must first take place [3]. The master node will broadcast a message. All nodes that receive the message will respond with a node ID. The master node will store this information. Each of these nodes will then broadcast their own messages, and receive back the IDs of all nodes that received that message. These IDs will also be sent to the master node. This process will be repeated until a routing table, consisting of each node and its neighbors (the nodes it can communicate with), is stored in the master node. This table will be used to create and update optimal communication routes from each node to the master node.

Another product of this process will be a field registration to determine the locations of each of the nodes. The communication process will be used to determine node locations relative to the field, and the Global Positioning System at the gateway nodes will allow absolute positions to be determined.

After the field is initialized, each time two nodes want to communicate with one another, they must follow a protocol developed for node to node communications [3]. This is in order to insure reliable communications between nodes. When one node wants to send a message to another, it first sends a *request to send* (RTS) message to that node. The RTS is sent at a nominal source level and at a lower bandwidth than a standard message. If the receiving node does receive the RTS, it replies with a *clear to send* (CTS) message at the same source level. If the originating node receives the CTS, it then sends its message to the receiving node at the same source level and at a higher bandwidth. If the originating node does not receive the CTS, it sends a new RTS at a higher source level. This process is repeated until it receives a CTS, at which time the message is sent at the successful source level. An error message will be sent back to the originating node if the message does not arrive at the receiving node intact.

This communication process is one of the main sources of energy consumption at the node. The other sources include the energy necessary to power the sensors and process the sensor data.

3 Field Optimization and Control

The goal of the DADS Network Control and Optimization task is to increase field lifetime by controlling power consumption while maintaining field level detection capability. For the purpose of this study, field lifetime was defined as the time until the first *n* nodes expend all of the energy. Four field functions were found to be amenable to control for the purpose of extending the lifetime of the field. These four functions were the node to node communications, the strategy for reporting potential detections, the processing mode of each sensor node (wake, sleep, etc.), and the routing of communication messages from each node to the master node. It was determined through preliminary studies that the control of communications routes had the highest potential for increasing the field lifetime, and was therefore chosen as the focus of this investigation.

3.1 Communication Network Routing

A major problem associated with the routing of communications in the DADS field is that nodes that relay a large number of messages will consume a large amount of energy. This will cause these nodes to die out much faster than the rest of the field, cutting much of the field off from communication with the master nodes, thereby leaving the field unable to meet performance requirements. In order to avoid this problem, a dynamic routing algorithm was created to increase system lifetime. The algorithm determines the optimal *routing strategy* for each time step. A routing strategy is the family of routes from each sensor to a master node.

Dynamic control of the DADS communications network consists of creation of the initial routing strategy, and the modification of the routing strategy as time progresses.

When the DADS field is initialized, a routing table will be produced that lists every node to which an individual node can talk. This table will be stored in the master node. From this table, the initial routing strategy will be determined. As time progresses, the routing strategy may need to change in order to prevent some portions of the field from burning out faster than other portions. The master node will maintain a database with estimates of each node's remaining energy and will periodically poll the routing algorithm to check if rerouting is in order. The algorithm will change the routing strategy only when it is determined that doing so will increase the field lifetime.

3.2 Modeling Field Functions

Several of the field processes have been modeled in order to develop a controller for the field. First, a very simple sensor performance model is used [4]. This model assumes "cookie cutter" detections. That is, each node is given a detection range. A target is detected with probability 1 by a sensor if the target is within the sensor's detection range, and detected with a probability of 0 if it is not. There is exactly one message sent to the master node for each detection.

The expected energy to send a message for every possible node to node communication path is pre-computed using the node-to-node protocol described in section 2. This calculation uses the *passive sonar equation* [3], given by

$$E_b/N_0 = SL - TL - AN + AG - 10\log_{10} b$$

where,

E_b/N_0 = energy per bit divided by noise power spectral density

SL = source level

TL = transmission loss

AN = ambient noise

AG = array gain

b = bandwidth.

Transmission loss is modeled as

$$TL(r) = 20\log r + \alpha r$$

where r is the range and α is the absorption.

Given the power P_0 required to send a message at a nominal source level, SL_0, the power P required to send a message at source level SL is

$$P = P_0 * 10^{(SL-SL_0)/10}.$$

The calculation of the expected energy for node A to send a message to node B is calculated using the following algorithm [4]:

```
Loop over all possible outcomes for the number of attempts to successfully detect the alert and the report messages
    Calculate the energy required to send the alerts and reports and receive the acknowledgements, $A_k$.
    Calculate the energy required to receive the alerts and reports and send the acknowledgements, $B_k$.
    Calculate the probability this event occurred, $p_k$, given by $p_k = \Phi((E_b/N_0)/\sigma)$.
```

The expected energy to send a message from node A to node B is $\sum p_k A_k$ and the expected energy to receive at node B is $\sum p_k B_k$.

The battery power model in the sensor node is a combination of a steady state power loss for in-node processing and the node-to-node communications model.

Another aspect of the field that was modeled was the expected message generation rate per node. The algorithm starts with an estimated message generation rate for each node. After each time period, the estimated message generation rate is updated by taking a weighted average of the short-term mean generation rate during the last time window and the long-term mean generation rate [4].

3.3 MOE Calculations

The MOE that this algorithm seeks to maximize is the expected remaining lifetime of the field. The models presented in the previous section will be used in this calculation.

Every time the master node polls the controller, the controller will generate a number of possible routing strategies, and will compute the expected remaining lifetime of the field for each strategy. First, the energy used to change from the current routing strategy to the candidate strategy will be calculated. The algorithm assumes that messages will only be sent to the affected nodes when a routing strategy is changed.

Next, the expected message rate and steady state energy loss are used to compute the expected hourly energy usage per node. This is then divided into the expected energy remaining after rerouting, giving an expected lifetime for each node for a candidate routing strategy.

The expected remaining lifetime of the field is the expected time until the first n nodes fail.

3.4 Dynamic Control Problem Formulation

The algorithm developed to determine the optimal routing strategy uses a one step rollout approach. The one step rollout algorithm, a simplified version of the Neural Dynamic Programming (NDP) approach, is an approach to stochastic control using dynamic programming [5]. The rollout algorithm seeks to minimize, over all possible control strategies, a *cost-to-go* function, which is the expected cost to termination from each state of the system. The cost-to-go is given by Bellman's equation,

$$J^*(i) = \min_u \sum_{j=1}^{n} p_{ij}(u)\left[g(i,u,j) + J^*(j)\right],$$

where

$p_{ij}(u)$ is the probability of transitioning from state i to state j given control strategy u,

$g(i,j,u)$ is the cost of transitioning from state i to state j given control strategy u,

and $J^*(j)$ is the cost-to-go from state j.

The rollout algorithm estimates the cost-to-go, $J^*(j)$, using a base heuristic.

In DADS, Bellman's equation has been altered to maximize the cost-to-go, which is the expected remaining lifetime of the field. The algorithm works as follows: an initial routing strategy is determined using either a minimum hop algorithm or the expected remaining lifetime calculation. When an update is requested the algorithm creates a large number of candidate routing strategies, including the current one, and calculates the expected lifetime for each candidate routing strategy. The lifetime is calculated assuming that the field will maintain this route for time T, and then revert to some base heuristic. The base heuristic used in this approach is to keep the current routing. The cost of rerouting is included in the expected remaining lifetime calculation. The routing candidate with the maximum expected lifetime is chosen, unless it fails to show significant improvement over the current route.

3.5 Control Strategy Selection

Even for a relatively small field, the total number of possible routes is quite large. In order to search for the route with the maximum expected lifetime, two techniques have been developed. The first technique attempts to intelligently prune away routing strategies that are unlikely to produce positive results. The pruning strategy employed by the control algorithm is simple [2]. If node A wants to send a message to one of the master nodes, the distance between node A and that master must fall beneath a certain threshold. Also, if node A wants to relay the message through node B, then the angle formed by connecting nodes A, B, and the master node must be within a predefined threshold. If no path is available that meets this criteria, these restrictions are relaxed until a path exists from node A to the master node. The family of routing strategies that survive this pruning process is exhaustively searched until the one with the largest expected remaining lifetime is found.

The other method uses genetic algorithms [6]. Genetic algorithms attempt to model the biological processes of natural selection, also known as "survival of the fittest", in order to reach an optimum.

Using the genetic algorithm, an initial "population" of entities, represented by their "chromosomes", is chosen. A "fitness function", which provides a quantitative measure of goodness, is also developed. Each of the entities in the initial population is evaluated using the fitness function. Crossover (sexual reproduction), cloning (asexual reproduction), and mutation are then performed on the current generation's population, in proportion to the fitness of each member of the population. This creates the new generation. This process is repeated until a stopping criterion is met.

For the purpose of this problem, the initial population of chromosomes is a list of candidate routing strategies. For each strategy, $\text{route}[j]=k$ indicates that node j sends its messages to node k, and $\text{route}[j]=j$ indicates that j is a terminal node. The initial population is filled out by randomly selecting minimum hop routes.

The fitness function is the expected remaining lifetime of the field.

The crossover process for DADS is complicated. Two parents are chosen using fitness proportional selection. These two parents will form two children. Initially, an empty adjacency matrix, which is essentially an empty routing table, is created for each child. For each node, one of the parents is randomly chosen, and the path from that node to the master for that particular parent is input into the adjacency matrix for child 1. The path from that node to the master for the other parent is entered into child 2. After this is done for each node, each child will have an adjacency matrix that will often contain multiple paths to the master node from some of the nodes. A minimal spanning tree of each child's adjacency matrix is used to determine the routing schemes. The children will then replace their parents in the next generation.

For cloning, routing strategies that were not chosen for crossover are copied into the next generation.

After a new generation is formed, each routing scheme will then go through the mutation process. For each routing scheme, for each node, a list of all the neighbors it is able to route through, other than the one it is currently routing through, is generated. If this is a non-empty list, the algorithm randomly decides whether a mutation should occur using the mutation rate. If mutation does occur, a node is randomly selected from the list and the current node is rerouted to it.

The stopping condition used for DADS was to run the algorithm for a predefined number of generations.

4 Tests and Results

Several scenarios were run using both the pruning and genetic algorithm controllers [2]. These scenarios were run on the Wagner Associates' developed DADS Module for Dynamic Network Control (DMDNC), a Monte Carlo simulation [4]. Several metrics were created to evaluate the performance of the controllers, as they were compared to some static routing strategies and to each other.

4.1 Simulation Environment

DMDNC is a Monte Carlo simulation that was developed to model the operational features of DADS that are amenable to dynamic control. For various dynamic control approaches, DMDNC can be used to evaluate the operational performance of a DADS network and calculate a set of metrics. The following is a list of the major components of DMDNC:

- Field of nodes
- Target motion
- Detection process
- Energy loss
- Routing
- Communications protocol
- Reporting strategy
- Sensor node processing mode

The last four items are controlled by the dynamic controller.

4.2 Metrics

There are a number of metrics generated by DMDNC to test the effectiveness of a controller. This paper will focus on three of them.

Proportion of nodes up as a function of elapsed time. The proportion of nodes up is the number of the nodes in the DADS field that are not dead divided by the number of nodes in the DADS field.

Proportion of detected opportunities as a function of elapsed time. The proportion of detected opportunities is the number of detections that have occurred since the beginning of the mission divided by the number of opportunities for detection since the beginning of the mission.

Proportion of reported opportunities as a function of elapsed time. A successful report is a message that reaches a master node. The proportion of reported opportunities is the number of successful reports that have occurred since the beginning of the mission divided by the number of opportunities for detection since the beginning of the mission.

4.3 Scenarios

The controller was tested against some static routing strategies. Table 4.1 gives some of the important field parameters for this scenario.

Parameter	Value
No. of nodes	36
Comm. Distance (m)	1000
Master node energy	2000 W-hrs
Sensor node energy	1000 W-hrs
Nominal source level (dB)	170
Power loss to transmit at nominal source level (W)	3.3
Power loss to receive	0.182
AN + AG (dB)	50
Initial source level (dB)	190
Increase in source level for successive transmission (dB)	10
Duration of mission	90 days
Route update freq	5 days

Table 4.1: Parameters for DADS Scenario

The field consisted of 32 sensor and 4 master nodes in a 55km X 55km square (See Figure 4.1). The starting energy was 1000 W-hours for the sensor nodes and 2000 W-hours for the master nodes.

There are 900 Monte Carlo tracks entering the area of interest from the west, spread uniformly over the 90-day mission. The targets travel between 8-10 knots and leave the area of interest along the eastern edge of the region. The detection range for the target detection model is 2 km.

This scenario was run using DMDNC and four different methods to calculate message routes. The first, titled "No Poll", used a shortest path algorithm to create a routing strategy on day 1, and maintained that same routing strategy throughout the life of the field. The second, titled "Init Poll", chose the strategy that had the greatest expected lifetime of the field on day 1, and maintained the same routing strategy throughout the life of the field. The third, "NDP", started with the same routing strategy on day 1 as Init Poll, but used the dynamic controller in pruning mode to check for alternative strategies every 5 days, and to change strategies when it was beneficial. The final strategy, "GA", started with the same routing strategy on day 1 as Init Poll, but used the dynamic controller in the genetic algorithm mode.

In order to create the initial generation for the genetic algorithm, 500 routing schemes were randomly generated using a shortest hop algorithm. The 100 "fittest" schemes were then chosen to comprise the initial generation. A crossover rate of 0.9 was used, meaning that 90% of a subsequent generation was formed by crossover, while 10% were carried over as is (cloned). The mutation rate was set at 0.1, meaning that for each routing scheme in a generation, there was a 10% probability for each node that the route would be mutated. The number of iterations was set at 20, meaning that the algorithm stopped after generation 20 was formed.

Figure 4.2 shows the *proportion up* statistic for the four methods. Notice that the nodes stay alive longer using the NDP and GA methods, but once they start to die, they do so at a much faster rate than the other methods. This is due to the fact that the controller seeks to spread out the energy consumption evenly. When the first node does die, many of the others are similarly low in energy, and die soon afterward.

A look at the *proportion detected opps* and *proportion reported opps* (Figures 4.3 and 4.4) graphs show that the dynamic control methods allow the field to be useful significantly longer than the other methods.

The two dynamic control methods performed almost identically. The genetic algorithm, however, with the parameters specified above, ran approximately an order of magnitude faster than did the pruning algorithm. For cases with a larger number of nodes, this difference should be even larger, given appropriate population sizes and number of iterations.

Two other scenarios were run which varied some of the field parameters. One reduced the number of master nodes from four to two, and the other made several changes in node energy levels, detection ranges, and other parameters. The basic results of these tests were similar to the first. NDP and GA produced similar results, and significantly out performed No Poll and Init Poll.

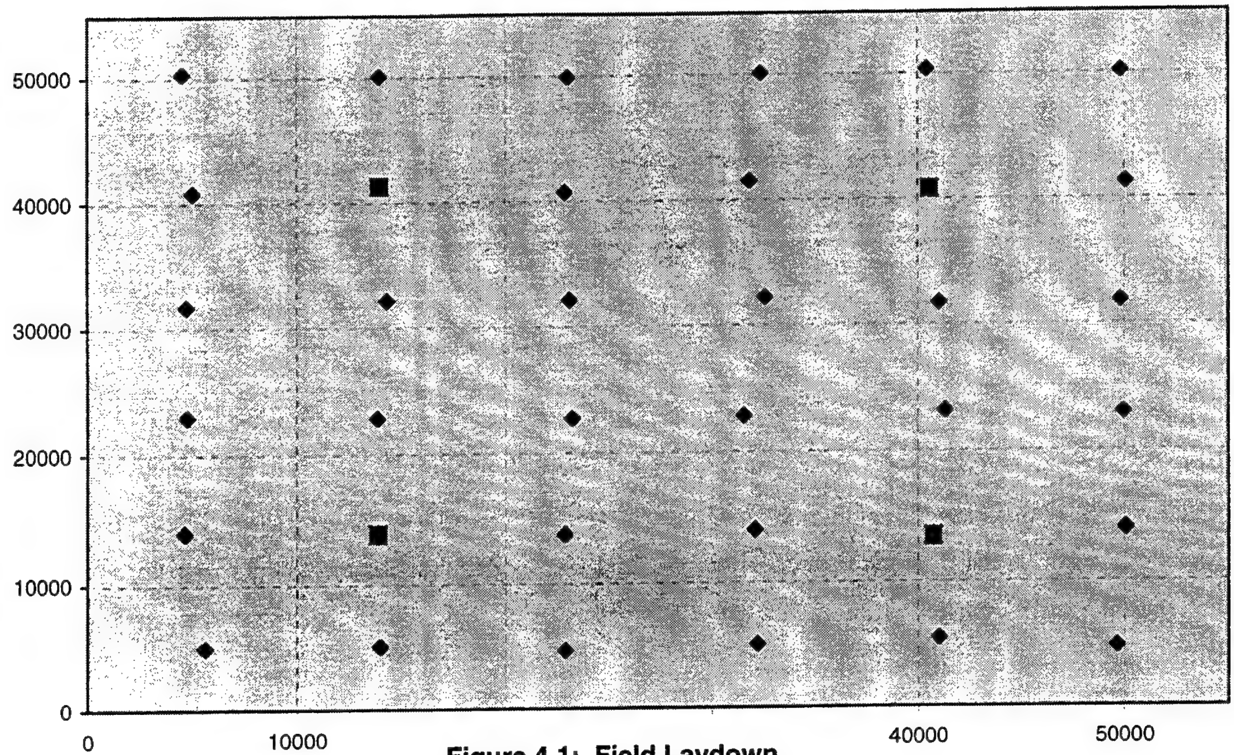


Figure 4.1: Field Laydown

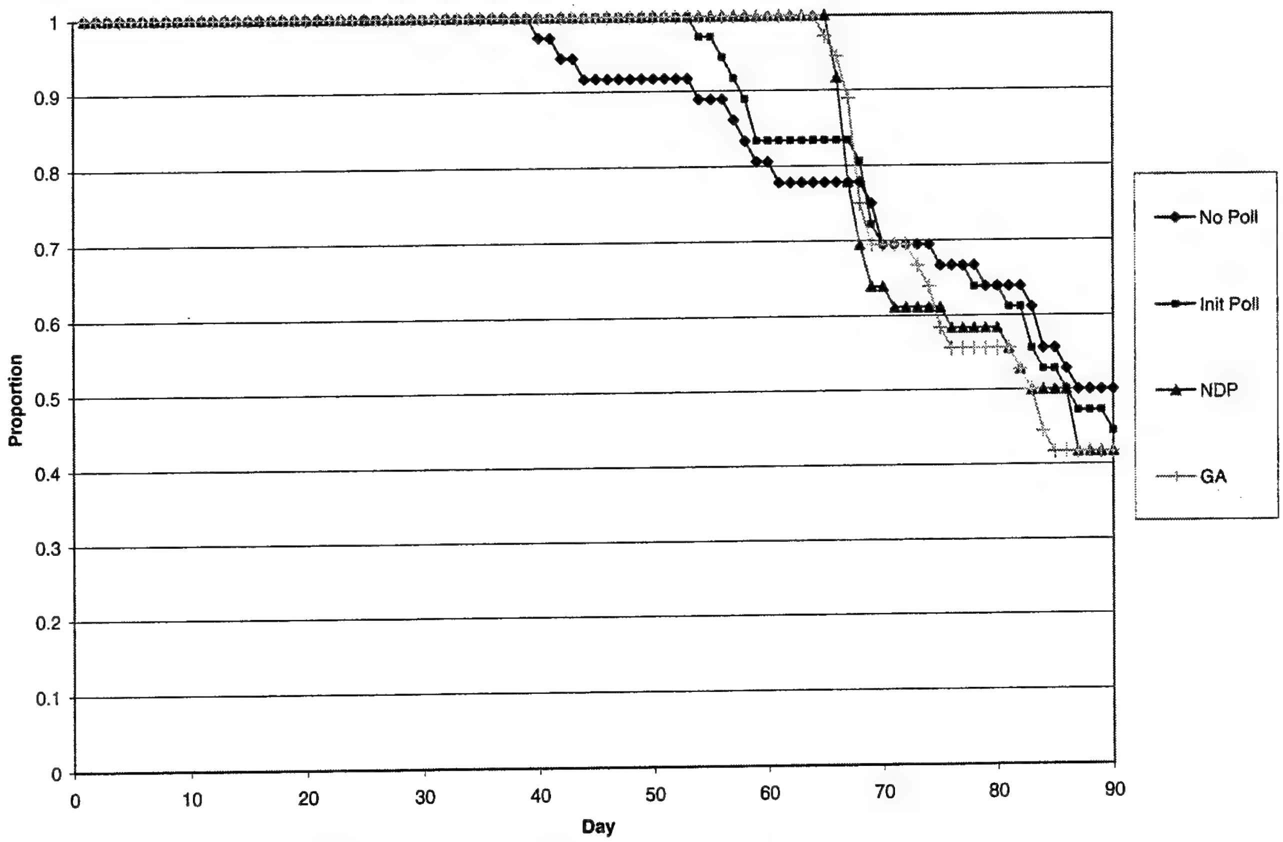


Figure 4.2: Proportion of nodes up as a function of elapsed time

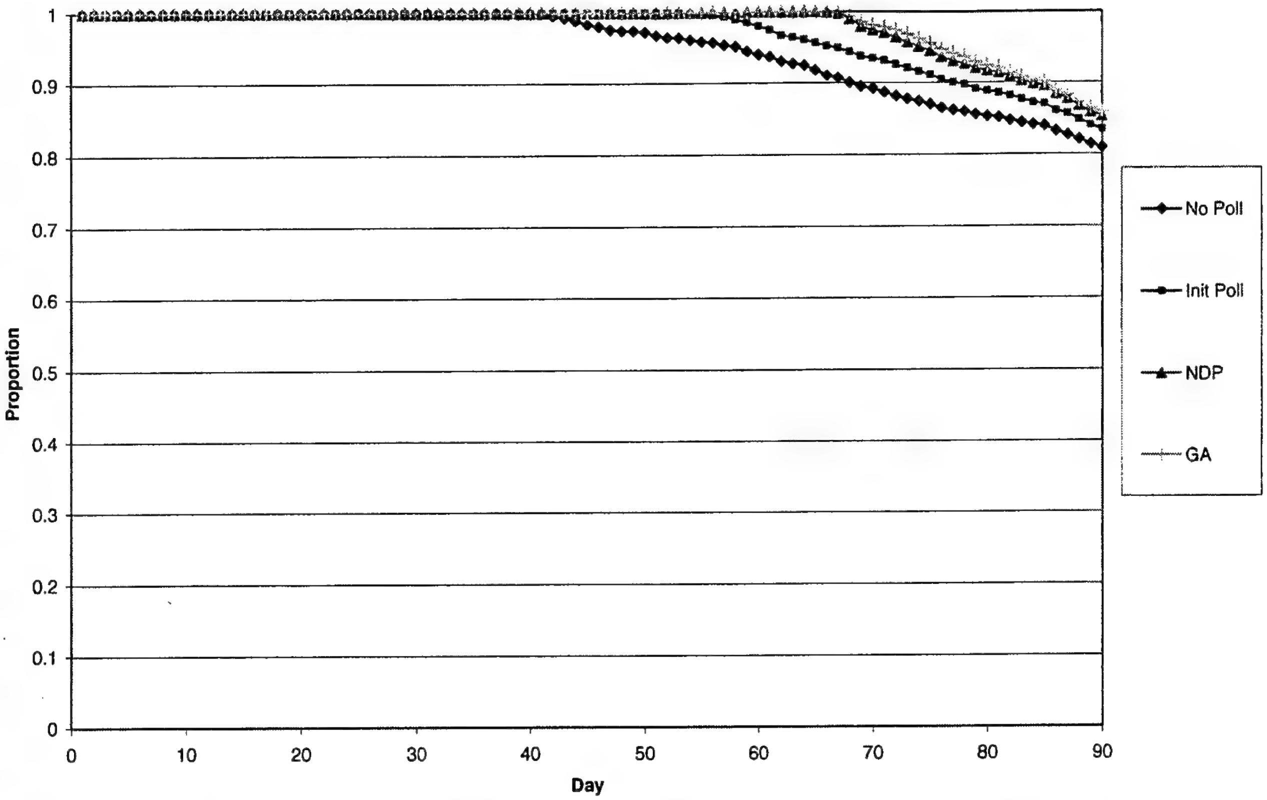


Figure 4.3: Proportion of detected opportunities up as a function of elapsed time

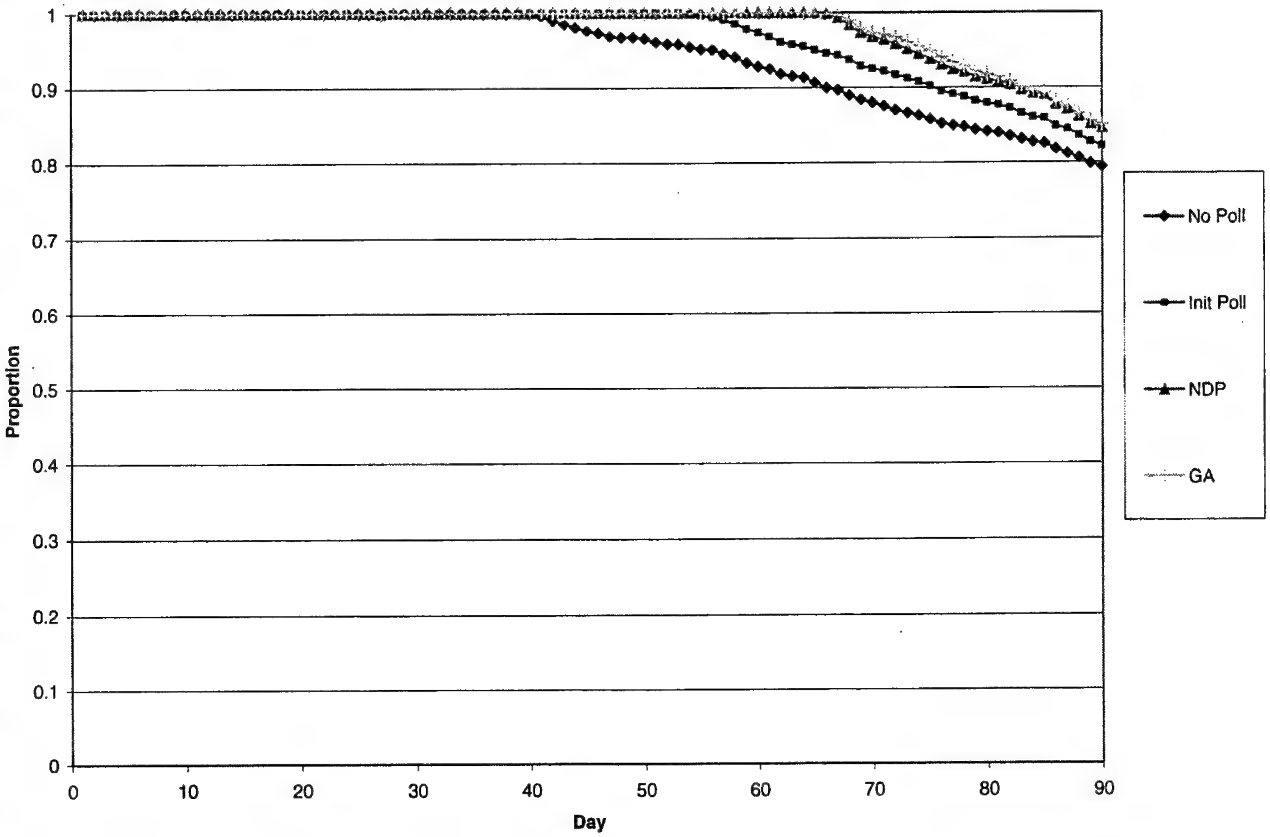


Figure 4.4: Proportion of reporting opportunities up as a function of elapsed time

5 Summary

From the simulations performed this study, one can conclude that intelligently controlling the communications network to maximize field lifetime shows great potential benefit to a DADS field. The rollout algorithm has proven very successful in initial tests, using both the pruning strategy and genetic algorithm to search for optimal routing strategies. As the genetic algorithm has proven to take less computation time, it is the preferred method for further development. Examination regarding actual real time implementation of these algorithms in an operational system may uncover issues yet unknown, but at the exploratory development phase of this effort, these two different methods show potential for application.

6 References

[1] Mark Hatch et al. *Data Fusion Methodologies in the Deployable Autonomous Distributed Systems (DADS) Project.* First International Conference on Multisource, Multisensor Information Fusion (Fusion 98), July 1998.

[2] Margaret E. Grunnert et al. *Dynamic Control for Deployable Autonomous Distributed System.* Final Report, Contract # N66001-98-C-6501, August 1999.

[3] J. A. Rice and M. D. Green. *Adaptive Modulation for Undersea Acoustic Modems.* Proceedings MTS Ocean Community Conference, Vol 2, pp 850-55, November 1998.

[4] Stanley J. Benkoski et al. *DADS Model for Dynamic Network Control.* Final Report, Contract # N66001-97-X-6009, June 1997.

[5] D. P. Bertsekas and D. A. Castanon. "Rollout Algorithms for Stochastic Scheduling Problems." *Heuristics,* 1996.

[6] David Edward. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Publishing Co., Inc. January, 1989.

Dynamic Information Dissemination for Control of Rapidly Changing Enterprise Systems *

Alvin S. Lim

Computer Sciences and Engineering
107 Dunstan Hall
Auburn University
Auburn, AL 36849
lim@eng.auburn.edu
phone: (334) 844-6326
fax: (334) 844-6329

Abstract

For large-scale enterprise systems to respond rapidly to dynamically changing situations, real-time information must be disseminated dynamically from mobile data sources through reconfigurable communication infrastructures to the components that control dynamic re-planning and re-optimization of the enterprise based on newly available information. Enterprise information systems may consist of very large number of highly mobile sensor sources and users scattered over a wide area with little or no fixed network support. Mobile transactions and query processing through this amorphous, fluid and unstructured networks of information sources and users must use an integrated approach to solve the problems of mobility, dispersion, weak and intermittent disconnection, dynamic reconfiguration, and limited power availability. Most mobile databases developed so far to address these problems duplicate the functionalities supported by the underlying mobile and wireless network infrastructure for solving these problems and supporting end-to-end mobile communication. Furthermore, they assume that mobile hosts and data sources are accessible from a traditional fixed well-structured computer networks through a single wireless hop. The solutions to these problems require a more integrated approach with efficient coordination and little duplication of functionalities between the three mobility-aware system layers: mobile information systems, configurable operating systems and network layers, and the physical mobile device layer.

1 Introduction

The adaptive control of dynamically changing enterprise systems will depend critically on real-time information gather from integrated low-powered sensors and mobile devices [10, 17, 16] deployed throughout the enterprise. These dynamic enterprise information systems will consist of very large number of highly mobile data sources and users scattered over a wide area with little or no fixed network support.. These mobile and miniaturized information devices, such as smart sensors and RFID tags, will be equipped with embedded processors and wireless communication facilities, information storage capability, smart sensors and actuators. The benefits are overwhelming since these devices will make information systems more intuitive, flexible, easy-to-use, low-maintenance, portable, ubiquitous, reliable and task-specific.

Unlike traditional well-structured computer networks, networks of embedded sensor devices used in dynamic enterprises are unstructured and very large, possibly in the order of tens or hundreds of thousand nodes in a localized area. Wireless mobile information nodes need to form temporary ad-hoc networks in lieu of any established infrastructure with centralized network administrator. Runtime facilities for information processing and communication must be capable of adapting to the following problems of amorphous networks. First, wireless communication in mobile information devices have limited range and bandwidth. They are much smaller in size and have limited capability, with range of limited processing capacity and memory storage. Second, both mobile data sources (servers) and mobile clients are highly

*This research is supported in part by the National Science Foundation under grant CCR-9896086.

mobile. Third, tens of thousands of mobile information devices, sensor nodes, mobile support hosts may be deployed in the field over a wide dispersed area. Large number of mobile RF nodes and sensor nodes must be used to relay information over a long distance to an access point in the fixed network. Fourth, no fixed network infrastructure exists for the large number of mobile nodes in an area, such as a remote distributed center, hazardous environment or legacy manufacturing facility that contain little infrastructure support and many uncertainties. Fifth, distributed mobile database applications must handle heterogeneity and very large number of different types of mobile information nodes and devices. Sixth, wireless communication links are subject to weak and intermittent connections and variability in bandwidth. Finally, since limited power is available in portable mobile devices, communication protocols must conserve battery energy.

The solutions to these problems require a more integrated approach with efficient coordination and little duplication of functionalities between the three mobility-aware system layers: mobile information systems, configurable operating systems and network layers, and the physical sensor layer. The mobile information layer contains mobility-aware mediators and adaptive sensor query processing. The runtime reconfigurable operating systems and network facilities contains mobile and adaptive protocols. The physical sensor and mobile devices layer handles the raw data, device presence detection and physical communication signals.

The main principle of our approach is that although mobility and wireless communication adversely affects all layers of the sensor information system, the overall system performs best when the information system at the higher level exploits functionalities implemented in the lower network layer. The different layers cooperate with one another to adapt quickly to changes in the underlying network structure and the availability (or mobility) of sensor and mobile devices. These changes can be detected most rapidly by the lowest physical device layer which will notify the adaptive network and reconfigurable operating systems layer. They will in turn notify the sensor information application layer of the changes. Upon notification of the changes, each layer would quickly adapt their operations to overcome problems caused by those changes.

2 Architecture of the MAIN

Critical real-time information are disseminated to various components in a dynamic enterprise through a Mobility-aware Amorphous Information Network (MAIN). These networks may be formed spontaneously and reconfigured dynamically when computing devices are deployed and when they move around. The mobile information system must be integrated with the mobile network system and be made aware immediately of the mobility and changes in the network environment. The architecture for MAIN is a synergy between three key mobility-aware system layers (Figure 1):

1. mobile information processing layer,
2. configurable operating systems and mobile network, and
3. physical mobile devices layer.

At the mobile information system level, the cooperative network of ***mobility-aware mediators*** and ***mobile device wrappers*** provides efficient access to diverse heterogeneous mobile data through the amorphous network. The mobile sensor information layer is supported by three major components: interoperable mobile object, dynamic query processing and mobile transactions. In the interoperable mobile object model, cooperative network of mobility-aware mediators and wrappers will be configured to support interfaces to remote mobile data sources through multihop wireless networks.

At the configurable operating system and mobile network level, adaptive network facilities provide runtime reconfiguration of amorphous mobile networks and reconfiguration notification to the mobile information system layer. Head mobile nodes of a cluster are controlled by embedded operating systems and network facilities that can be reconfigured at runtime to access the sensor and mobile devices in the cluster. When mobile devices move into a cluster, they register with an agent in the current network. Both mobile devices and head mobile nodes may move independent of each other.

At the physical level, different physical sensor and mobile devices may be assembled impromptu or reconfigured dynamically. The amorphous information network consists of large varieties of physical devices and computers, such as micro-sensor devices, larger mobile sensor nodes, RF relay links, palmtop and laptop computers, interrogator, desktop computers and communication processors. In this mobile network environment, we classify them under four node types: (1) *Sensor devices:* smart micro-sensor devices and small RFID tags, (2) *head mobile nodes:* large mobile devices, such as large tag devices, hand-held interrogators, palmtop and laptop computers, (3) *base nodes:* desktop workstations directly connected to

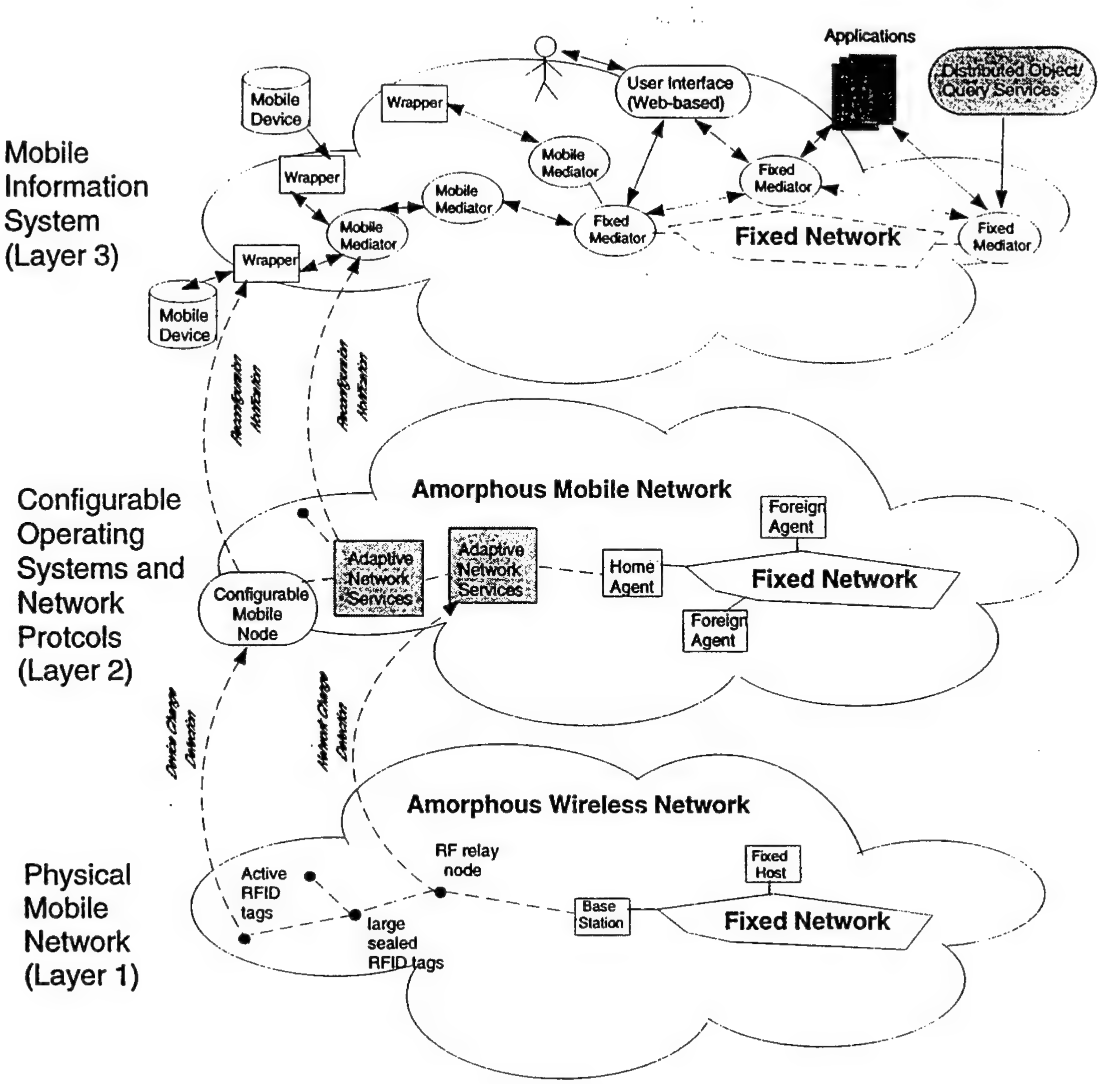


Figure 1: Amorphous Mobile-Aware Layered Architecture

a fixed network that have wireless links to mobile nodes, and (4) *fixed nodes:* larger desktop workstations connected only to a fixed network, but no wireless links to mobile nodes. Each of the above four types of nodes contains the software components for two main software layers – mobile information processing and configurable operating system and network facilities.

3 Interoperable Mobile Object Model

In the interoperable mobile object model, mobile information clients in a dynamic enterprise may access and update mobile data sources in the enterprise through a group of mobility-aware mediators, object servers and mobile device wrappers that coordinate together using multi-hop wireless and mobile networks (Figure 2). Mobile network protocols, such as dynamic source routing [8] and Mobile IP [12], are responsible for dealing with the problem of mobility by treating it as a routing problem. In order to reduce communication cost due to tunneling and improve performance, mediators and object servers must also be aware of the mobility of mobile data sources and cache location binding information. Mediators and object servers may themselves be implemented in mobile hosts. Mobile end-users and applications pose queries to the mediators, which coordinate among themselves to decompose, schedule and route queries to the mobile data sources through wrappers. The mediator resolve the bindings of the mobile data sources through the object servers. The binding maps the object unique identifier to wrappers

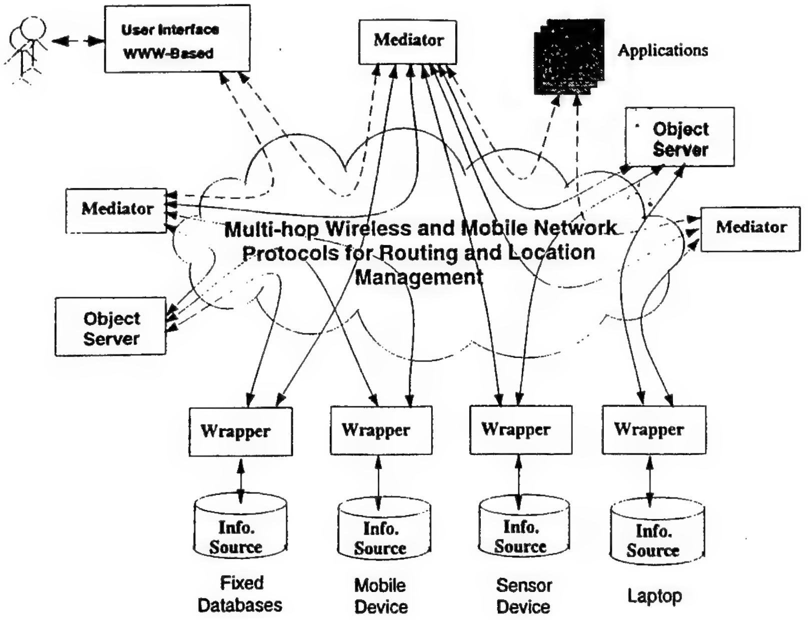


Figure 2: Wireless and Mobile Network Support for Coordination among Mobile-Aware Mediators, Object Servers and Wrappers for Mobile Data Sources

at the mobile devices specified by their IP address, port number, and segment ID within the mobile device. Once the binding is resolved, the mediator communicate with the mobile device directly. When the mobile device moves to a new location, Mobile IP protocol tunnels the message datagrams from mediators to the mobile device's new location. With mobile IP optimization, address caches at the mediator host will enable future messages from the mediators to be transmitted directly to the mobile host's new location without tunneling.

3.1 Mobility-Aware Mediators

Mediators cooperate with each other to play the key roles in dynamic query processing and mobile transaction of dynamic enterprises. When a mediator receives a query, it may decompose it into multiple subqueries and forward them to the appropriate mediators that are associated with mobile data sources of each sub-query. The mediator that first receives the query selects other mediators for each sub-query based on its knowledge of the current location of the mobile data sources involved in the query. Dynamic query processing are performed within mobile transactions. Each mediator maintains an information consumer's domain model and many sensor information producer's source models. While the initial mediator is responsible for the overall transaction, the subsequent mediators are responsible for the sub-transactions and their related locking and logging functions. Mediators are aware of mobility and wireless network conditions through notification from the lower network layer. To access mobile information sources from mobile devices, mobility-aware mediators can reconfigure the routing and scheduling of sub-queries to different location or mobile objects in response to mobility, disconnectness, and bandwidth variability.

3.2 Mobility-Aware Object Server

In highly mobile enterprises, object servers support mobility of data sources and are responsible for (i) mobility registry, containing information on names, addresses, port and segment ID binding as well as the current location of mobile data sources which have adopted the object server as its home, (ii) wireless condition cache that stores information on connectivity and quality of links to mobile data sources, and (iii) replicated data repository, which contains the sets of object servers that replicate the mobile device information.

Each mobile data source has exactly one *home object server* at any time, although its home object server may change if the mobile device has changed its location for an extended time and adopted another home object server. A home object server has the same network number as the mobile host. Home object servers are responsible for caching information from mobile data sources. When a mediator request for a particular mobile data source from an object server, the object server will return the address and port of the wrapper of the mobile device. Subsequently, the mediator communicates with the mobile device directly and does not use the object server.

If the mobile device has moved to a new location, either dynamic source routing or Mobile IP will be responsible for tunneling messages (or rediscover the route) from the mediators through the home agent to the new location. When the mobile device has moved to a new location, Mobile IP registration process will forward the new care-of address to the home agent in the mobile device home network. Through location binding update similar to optimization in Mobile IP, the object server will also obtain the new care-of adress. The object server caches this care-of address and will use it for future binding request. When a mobile device moves to a new location, the reconfigurable network facilities notify its home object server and mediators that are communicating with the mobile device of its new location. If the mobile device has moved to a different part of the network, the mediator may re-evaluate the query decomposition and use alternate query routing and scheduling of updates and queries to mobile sensor data sources. Early detection and notification of mobile device location by the physical network improves the performance of distributed queries through updated query routing and scheduling that reflects new mobile device locations. Our object server is similar to object repository in Thor [2] in that each mobile object has exactly one object server. However, we use a different approach, in that unlike Thor, mobility of objects is handled primarily by Mobile IP or an ad-hoc routing algorithm. Object servers may cache location binding information for performance enhancement.

3.3 Mobile Device Wrapper

For spontaneously assembled enterprises, software wrappers enable incrementally deployed ad-hoc components to interoperate with each other. Wrappers are software modules, each serving one mobile data source. In order to make an existing mobile information source available to the network of mediators, a wrapper is built around the existing mobile device to turn it into a local agent for the mobile object. The local agent is responsible for accessing mobile information source and obtaining the required data for answering the query. Wrappers are customized to integrate with techniques useful for mobile nodes, such as autonomous identification and location management. Services provided by a mobile device wrapper also include translating a subquery in consumer's query expression into an mobile information producer's query language expression, submitting the translated query to the target information source, and packaging the subquery result into a mediator object.

The wrapper is also responsible for local management for data stored in the mobile device. It contains the local locking mechanisms for the global concurrency control scheme and the local recovery mechanism based on write-ahead logs. For larger mobile devices, these mechanisms and their related state information are implemented within the mobile device. For micro mobile devices, these mechanisms may be implemented in head mobile nodes of a cluster with which the mobile devices are currently associated.

Specialized mobile device wrappers may be developed for each mobile data source, e.g. remote smart sensors, radio identification tags, large tags, interrogators, pocket PC, and wearable computing devices [14]. Only one such wrapper would need to be built for any given type of sensor information source (e.g., raw sensor data, flat ASCII files, relational, object-oriented, or HTML files). They convert data in raw format to interoperable mobile object format, turning each device into a mobile agent for the interoperable mobile object. In order to track these mobile agents and changes in the remote mobile device, the specialized mobile wrappers inform mediators of changes in locations and connection variables, such as disconnection, changes in bandwidth and error rates.

4 Mobile Transaction

Complex dynamic enterprises use mobile transactions to preserve higher level of consistency in spite of failure and mobility. A mobile transaction may be instantiated from a fixed or mobile host. It may involve query on fixed or mobile objects. For instance, a mobile client may pose a query on multiple mobile data sources. When a mobile user wants to instantiate a transaction, it first perform a lookup for a mediator which will decompose, schedule and optimize the query in a mobile transaction. The mediator may route sub-queries to other mediators which will obtain the address and location binding of the target mobile object from the object servers. Sub-queries are then routed directly to the mobile objects at their current locations. If the object server do not contain the current location, Mobile IP will tunnel the sub-queries to the current location if the mobile object is reachable. In ad-hoc networks, the object server may rediscover the route to the mobile device and stores it in its local cache. When a mobile device is disconnected, queries and updates may be performed on the object server that contains a replica of the mobile object's data. Upon reconnection of the mobile device, the results in the replicas will be merged with the mobile device where conflicts are resolved.

The system handles both the mobility of users and mobility of data sources using the mechanisms for supporting different characteristics of each case as described below.

4.1 Mobile User

We consider the mobile environment where the mobile user may reach the fixed network through a base station either in a single hop or over multiple wireless hops. First we consider the single hop case. When the user initiates a query through a transaction, the query is forwarded at the network level by Mobile IP through the base station to a mediator. The mediator then schedules and routes the query to the data source. When the user moves to another cell during the transaction, its mobile host registers itself in the new cell at the Mobile IP level through a foreign agent. Mobile IP will then tunnel all responses to the query and other subqueries through the base station in the new cell. The transaction information remains in the mediator and need not be transferred to another location while the mobile user moves. By allowing Mobile IP to handle mobility and not duplicating the operation at the transaction level, this scheme avoids the costly operation of transferring transaction information from base station to base stations as the mobile user roams around, as in the kangaroo transaction scheme [3].

In multi-hop networks, the route error maintenance and route discovery protocols will be initiated by network protocols of the mobile user host whenever it moves to a different location. New routes are cached in the network layer and mobile host may continue functioning without being aware of the mobility. However, the mobile hosts are made aware of the mobility so that they can decide when to re-route queries to different mediators to improve performance.

4.2 Mobile Data

A different scheme is used to handling mobility of transactions where sub-queries are sent to mobile data sources. Again, we consider the environment where the mobile data sources may be reached from the base station through a single wireless hop or multiple wireless hops. We first consider the single hop case. When a mobile data source is relocated, its mobile device will register itself in the new location using the Mobile IP registration protocol. Queries sent to the mobile device home address will be tunneled using Mobile IP to the new care-of address of the mobile data sources. In larger mobile device where local locking and recovery mechanisms are resident in mobile device, transaction information in the local data manager of the wrapper for the mobile objects need not be moved when the mobile data source moves. In smaller mobile device where the local locking and recovery mechanisms are in a different mobile nodes, transaction information must be moved to a new mobile cluster head node if the mobile device moves to a new cluster.

In multi-hop networks, the route error maintenance and route discovery protocols will be initiated by network protocols of the mediator host whenever the mobile devices move to different location. New routes are cached in the network layer and mediator may continue functioning without being aware of the mobility. However, the mediators are made aware of the mobility so that they can decide when to reschedule and re-route queries to different mediators or location to improve performance.

5 Information Dissemination in Dynamically Changing Enterprises

Large-scale dynamic enterprises, such as flexible manufacturing and military command and control, involve dynamically changing structures and control. Various components and servers work together to facilitate dynamic changes in the enterprises based on new real-time feedback information disseminated from numerous mixed types of mobile sensors and fixed data sources. This enables the enterprise to maintain crucial information for controlling interaction between components in the enterprise.

Information dissemination in highly dynamic enterprises may involve transmission of feedback data from sensors to the scheduler and from controllers to the actuators. At any time, components may join (or leave) the enterprise and be automatically connected (or disconnected) to the clusters and communication infrastructure. Once connected, they may interact with other components in the enterprise to perform coordinated tasks by gathering information from the information network and propagating useful information to other parts of the enterprise. This ad-hoc communication infrastructure supports mobile object server and mediators that maintain and disseminate current information. Rapid dissemination of these real-time information is critical for dynamically controlling the behavior of components and clusters in dynamic enterprises.

6 Comparison with Other Work

Many mobile information systems [6] has been developed to address various problems of mobility and disconnection at different software levels. They differ in their assumptions about the underlying mobile environment and available network infrastructures. In the Bayou mobile storage system [15], the mobile computing environment allows collaborative applications to read and write on shared databases in disconnected mode based on tentative execution of writes and a primary commit scheme. When the mobile hosts are reconnected, the Bayou system provides automatic conflict detection and supports application-specific merge procedures for conflict resolution. Schemes for managing mobile object location [13, 2] include caching of location information in the home and visitor location registries, replication of user profiles and working set, and object pointer forwarding. Mobile transactions, such as Kangaroo transactions [3], preserves atomicity, concurrency and recovery properties in the presence of mobility of data clients by splitting transactions and migrating subtransactions information from one base station to another as the mobile hosts move through the cells. Various data caching strategies have been used to enhance the performance of data access in disconnected and weak connectivity modes. Coda [11] uses system level techniques for hoarding, emulation and reintegration. Callbacks [11, 4] are used to notify clients of updates of files by another client or other changes in the system environments. Rover [9] provides useful system mechanisms for supporting mobility, such as queued RPCs, relocatable dynamic objects and object caching. WebExpress [5] also shows improved performance with file caching in web access over wireless networks. For different host mobility behavior, whether the mobiles hosts are more often connected or disconnected, different cache invalidation strategies may be more appropriate [1, 7].

This research develops an mobility-aware amorphous network to support mobile information system for dynamically changing enterprises through the following facilities: First, mobile-aware mediators, object servers and specialized mobile device wrappers supports mobile devices that are of small sizes and limited capability through object servers that may store replicas of mobile device information. Second, Mobile IP and ad-hoc routing protocols support mobility of mobile data sources and mobile users where the home agents updates the new address of the mobile devices through the foreign agents. The home agents tunnel queries and replies to mobile devices in their new location. Third, the reconfigurable network protocols allow mobile nodes to detect the changes in configuration of mobile device, weak connectivity and mobility of mobile nodes. The network layers notify the other layers, such as mobile transaction and dynamic query processing, of changes in the mobile nodes.

To support mobility more efficiently, there needs to be interaction not only between the system mechanisms, the middleware facilities and the applications, but also greater interaction between the network protocols and these other software layers. The integration and coordination between the three layers – mobile information systems, configurable operating systems and network layers – will reduce duplication of mechanisms to support mobility, which can be supported more efficiently at the lower facilities, such as the network and system layers. The main idea is that the lower layers can detect more rapidly the changes in the mobile information network structure and the availability (or mobility) of sensor and mobile devices. After managing these changes at the network and system level, the corresponding mechanism may notify the higher level mechanisms of the changes, such as location and disconnection. Mobility and weak connections can also adversely affects the functionalities of the higher layers, such as dynamic query processing, which may use the change notification from the system or network layer. For example, the mediators may receive the notification and re-route sub-queries in response to relocations of mobile hosts. In practice, these changes detection can be most rapidly by the lowest physical device layer which should notify the adaptive network and reconfigurable operating systems layer when changes occur.

7 Conclusions

The control of large-scale dynamically changing enterprises depends critically on efficient dissemination of real-time information from the mobile information sources to the control servers and from control servers to the actuators. The communication infrastructure used in these dynamic enterprises is usually ad-hoc, mobile and reconfigurable. We have presented an overview of the mobile information systems that cooperates with mobile network layers for support on mobility and notification of changes in the amorphous network. This approach use an integrated approach to avoid duplication of functionalities and enhance coordination between the three mobility-aware system layers: mobile information systems, configurable operating systems and network layers, and the physical sensor layer. We assume that communication between wireless and mobile hosts and data sources may require multiple wireless hops. Distributed mobile-aware mediators provide mechanisms for location management, identification, mobility, discovery of information sources, mobile transactions, and dynamic query processing.

References

[1] Barbara, D. and T. Imielinski, "Sleepers and Workaholics: Caching Strategies in Mobile Environment," *Proc. ACM SIGMOD*, MAy 1994, pp. 1-12.

[2] M. Day, B. Liskov, U. Maheshwari, and A. Myers, "References to Remote Mobile Objects in Thor," *ACM Letters on Programming Languages and Systems*, March 1994.

[3] M. Dunham, et. al., "A Mobile Transaction Model that Captures Both the Data and Movement Behavior," *ACM Mobile Network and Applications*, V. 2, N. 2, Oct 1997.

[4] J. Flinn, et. al., "Energy-aware aAdaptation for mMobile Applications," ACM Symposium on Operating Systems Principles, South Carolina, Dec 1999, pp. 48-63.

[5] B. Housel, et. al. "WebExpress: A Client/Intercept Based System for Optimizing Web Browsing in a Wireless Environment. *ACM/Baltzer Mobile Netowkring and Applications (MONET)*, 1997.

[6] T. Imielinski, H. Korth, eds., "Mobile Computing," Kluwer Academic Publishers, 1996.

[7] Jin Jing, et. al., "Bit-Sequences: An Adaptive Cache Invalidation Method in Mobile Client/Server Environment," *ACM Mobile Network and Applications*, V. 2, N. 2, Oct 1997.

[8] D.B. Johnson and D. Maltz, "Dynamic Source Routing in Ad-Hoc Wireless Networks," *Mobile Computing*, T. Imielinski, H. Korth, eds., Kluwer Academic Publishers, 1996.

[9] A. Joseph, et. al., "Rover: A Toolkit for Mobile Information Access," ACM Symposium on Operating Systems Principles, Colorado, Oct 1995, pp. 156-171.

[10] J.M. Kahn, et. al., "Next Century Challenges: Mobile Networking for Smart Dust," *ACM Mobicom*, 1999.

[11] L. Mummert, et. al., "Exploiting Weak Connectivity for Mobile File Access," ACM Symposium on Operating Systems Principles, Colorado, Dec 1995, pp. 143-155.

[12] C. Perkins, "Mobile IP," Addison-Wesley, 1998.

[13] E. Pitoura, G. Samaras, "Data Management for Mobile Computing," Kluwer Academic Publishers, 1998.

[14] A. Smailagic and D. Siewiorek, "The CMU Mobile Computers and Their Application for Maintenance," *Mobile Computing*, T. Imielinski, H. Korth, eds., Kluwer Academic Publishers, 1996.

[15] D. Terry, et. al., "Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System," ACM Symposium on Operating Systems Principles, Colorado, Dec 1995, pp. 172-183.

[16] The Ultra Low Power Wireless Sensors project. `http://www.-mtl.mit.edu/ jimg/project_top.html`

[17] The WINS project. `http://www.janet.ucla.edu/lpe.lwim/`

Invited Talk 3

Unavailable

Section 3

Adversarial Games: Models & Solutions

Dynamic Programming Methods for Adaptive Multi-platform Scheduling in a Risky Environment[1]

Dimitri P. Bertsekas
Dept. of Electrical Engineering and Computer Science,
M.I.T., Cambridge, Mass., 02139

David A. Castañon
Dept. of Electrical and Computer Engineering,
Boston University, Boston, Mass., 02215

Michael L. Curry, David Logan, Cynara Wu[2]
ALPHATECH, Inc., 50 Mall Road, Burlington, MA 01803

Abstract

In this paper, we investigate alternatives to simulation-based approximate dynamic programming methods for adaptive multi-platform scheduling in a risky environment. In a recent effort, we considered rollout algorithms, in which on-line simulation was found to be more reliable than off-line training. Unfortunately, a large amount of computational resources was required to run even a modest number of Monte Carlo simulations. In this paper, we consider alternatives to using simulation. The first approach consists of using limited lookahead policies, which reduce computational requirements by considering value explicitly over a limited horizon and approximating the value of the remaining stages. The second approach decomposes the problem into sub-problems corresponding to platforms. In our computational experiments, we found that many of the variations of these approaches required significantly less computation time than rollout algorithms and also obtained results that were substantially superior.

1. Introduction

The planning and execution of multiple missions in the presence of risk is a problem that arises in many important military contexts. In data collection applications, multiple UAV platforms may be tasked to interrogate different areas, with the risk of platform destruction as each platform pursues its collection mission. In attack air operations, multiple platforms follow risky trajectories to attack enemy targets. For both applications, sensors and communication equipment can provide up-to-date information concerning individual mission and platform status, and thus provide notification of platform losses. This creates opportunities for replanning, using feedback to retask surviving platforms in order to best achieve mission objectives.

In mathematical terms, the above class of problems can be formulated as Markov decision processes. At each stage of the process, decisions are made that affect the evolution of a system state, which is also influenced by random discrete events. The goal is to select the current decision as a function of the current state in order to optimize mission performance.

The principal approach for solving Markov decision problems is dynamic programming (DP). In comparing the available controls at a given state *i*, DP considers the current stage value, but also takes into account the desirability of the next state *j*. It "ranks" different states *j* by using, in addition to the current stage value, the optimal value (over all remaining stages) starting from *j*. This optimal value is denoted $J^*(j)$ and referred to as the optimal *value-to-go* of *j*. Unfortunately, it is well known

[1] This work was supported by the Air Force Office of Scientific Research under contract #F49620-98-C-0023.

[2] *Corresponding Author: phone (781)273-3388, fax (781)273-9345, e-mail cynara.wu@alphatech.com*

that the computation of J^* is overwhelming for many important problems.

There has been a great deal of research on DP methods that replace the optimal value-to-go $J^*(j)$ with a suitable approximation for the purpose of comparing the available controls at each state. These methods are collectively known as neuro-dynamic programming (NDP). Previously, we applied a particular class of NDP algorithms, known as rollout algorithms, to risky multi-platform planning and scheduling problems. Rollout algorithms are a form of NDP that exploit knowledge of suboptimal heuristic decision rules to obtain approximations to the optimal value-to-go. We developed several rollout algorithms for risky multi-platform scheduling, using on-line Monte Carlo simulations to evaluate the reference base heuristic policies, and found that they performed significantly better than the base policies as well as off-line training methods. However, even using a modest number of Monte Carlo simulations resulted in large computation times.

In this paper, we consider alternatives to using on-line simulations. In particular, we consider two approaches that use analytic approximations of the value function. We first consider a class of approximation techniques in which the control exercised at a state i is determined by considering the costs accumulated over several stages, and then applying an approximation to the value-to-go from the resulting states. The rollout algorithms considered in our previous effort are a special case in which a single-stage policy is employed and on-line simulation is used in combination with a base heuristic to approximate the value-to-go.

Our second approach involves exploiting the structure of the problem and decomposing the problem into sub-problems, each of which is associated with a corresponding platform. Each sub-problem is solved independently but takes into account the results of previously solved sub-problems.

The paper is organized as follows. In Section 2, we describe the data collection problem which we are addressing. In Section 3, we present the framework for limited lookahead policies. In Section 4, we describe our decomposition approach to the problem. In Section 5, we present some computational results.

2. Example Data Collection Problem

The graph in Figure 1 is an example corresponding to a data collection problem. Each node represents a geographical area of interest with a one-time value (i.e., data may only be collected once from each location). The arcs represent connectivity among the geographical regions and may be successfully traversed with a known probability. Platforms traverse the graph and collect data (value) at each node, or else they are destroyed while traversing specific arcs. If a platform is destroyed on an arc, the value of the destination node is not collected, which can result in retasking other platforms.

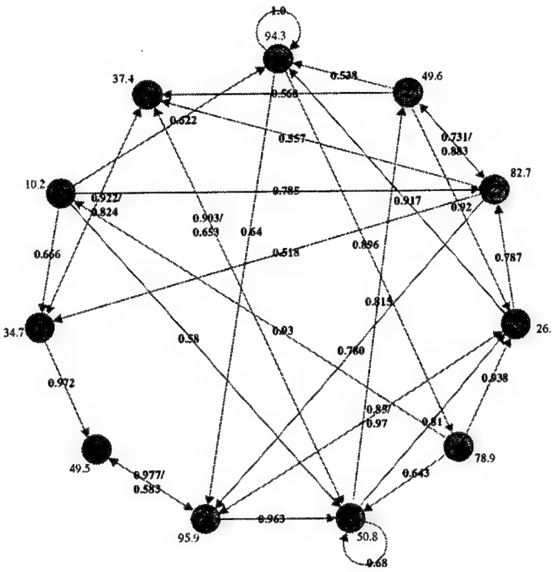


Figure 1 Graph Representation of the data collection problem.

The objective is to control the platforms in order to maximize the expected total value collected after N stages. Each platform begins at a base node (in this case, node 0 for all platforms) and may traverse one arc during each stage. There is a reward for each platform that has safely returned to its base node at the end of the Nth stage.

3. Limited Lookahead Policies

Consider a discrete-time dynamic system,

$$x_{k+1} = f_k(x_k, u_k, \omega_k),$$

where x_k is the state, u_k is the control to be selected from a finite set $U_k(x_k)$, and ω_k is a random disturbance. Denote the single-stage reward of control u from state x and disturbance ω by $g_k(x,u,\omega)$. A control policy $\pi=\{\mu_0,\mu_1,\ldots,\mu_{N-1}\}$ maps, for each stage k, a state x_k to a control value $\mu_k(x_k)\in U_k(x_k)$. There is a terminal reward $G(x_N)$ that depends on the terminal state x_N. The value-to-go of an optimal policy $\pi^*=\{\mu_0^*,\mu_1^*,\ldots,\mu_{N-1}^*\}$ starting from a state x_k at stage k can be computed using the following DP recursion

$$J_k^*(x_k)=\max_{u_k\in U_k(x_k)} E\{g_k(x_k,u_k,\omega_k)+J_{k+1}^*(f_k(x_k,u_k,\omega_k))\},$$

for all k and with the initial condition

$$J_N^*(x_N)=G(x_N).$$

For our problem, the state can be represented by a vector indicating for each node whether or not its value has been collected and by another vector indicating for each platform whether or not it is alive and if so, the node at which the platform is located. The control at a particular stage provides for each platform that is alive a node that the platform is to attempt to visit during the current stage. If the platform successfully traverses the arc connecting its current node to the next node and the value of the node has not yet been collected, the current stage reward includes the value of the node. If the platform successfully reaches its base node during the last stage, there is a terminal reward associated with the platform.

Under a one-step lookahead policy, the control selected at stage k and state x_k is that which maximizes the following expression:

$$\max_{u_k\in U_k(x_k)} E\{g_k(x_k,u_k,\omega_k)+\tilde{J}_{k+1}(f_k(x_k,u_k,\omega_k))\},$$

where $\tilde{J}_{k+1}$ is some approximation of the value-to-go function J_{k+1}^*. Under a two-step lookahead policy, the control selected at stage k and state x_k is that which maximizes the above expression when $\tilde{J}_{k+1}$ is itself a one-step lookahead approximation; i.e., for all possible states $x_{k+1}=f_k(x_k,u_k,\omega_k)$, we have

$$\tilde{J}_{k+1}(x_{k+1})= \max_{u_{k+1}\in U_k(x_{k+1})} E\left\{\begin{matrix} g_{k+1}(x_k,u_k,\omega_k)+ \\ \tilde{J}_{k+2}(f_{k+1}(x_{k+1},u_{k+1},\omega_{k+1})) \end{matrix}\right\}.$$

Other multi-stage lookahead policies are similarly defined. Note that the number of lookahead stages, M, should be less than or equal to N-k-1. Essentially, the M-stage lookahead policy selects at stage k its decision by determining the optimal policy if there were only M stages remaining and the terminal cost was given by $E_{x_M}\{\tilde{J}_{k+M+1}(x_M)\}$, where x_M is the state resulting from applying the policy for the M decisions. A decision is selected, and the process is repeated at the next stage. The lookahead horizon is limited to the number of remaining stages, and so if the number of remaining stages is less than M, the M-stage lookahead policy determines the optimal strategy. A special case of such policies in which the value-to-go is approximated with zero is referred to in the literature as rolling or receding horizon procedures.

Generally, the effectiveness of limited lookahead policies depends on two factors:

1. The quality of the value-to-go approximation – performance of the policy typically improves with approximation quality.
2. The length of the lookahead horizon – performance of a policy typically improves as the horizon becomes longer (at least for small horizon lengths, e.g., 1-4).

However, as the size of the lookahead increases, the number of possible states that can be visited increases exponentially. To keep the overall computation practical, the complexity of the value-to-go approximation should be reduced for larger lookahead sizes. Balancing such tradeoffs is therefore a critical element in determining the size of the lookahead and the method for approximating the value-to-go. This paper explores several possibilities and tries to quantify the associated tradeoffs. One of the advantages of using limited lookahead policies for our particular problem is that the number of controls at a particular stage is fairly small and as a result, the computation required to explore all states that can be visited over the next M stages is manageable for small M.

3.1. Pruned Limited Lookahead Policies

Since the number of states that can be visited over M stages grows exponentially in M and also in the number of platforms, limited lookahead policies for M>1 are impractical for problems with many platforms. One approach to reducing the computation required for limited lookahead policies is to limit the number of states that can be visited. This can be accomplished by "pruning" controls that yield inferior intermediate values.

A pruned version of a limited lookahead policy depends on an integer parameter B that is typically selected through trial and error. In particular, we determine the one-step lookahead values for all controls available from our initial state. Controls that are not among those with one of the B best one-step lookahead values are pruned. We then repeat this process for each state that can be reached from a control that was not pruned and determine the one-step lookahead values for all controls available from these states. For each of these states, controls that are not among those with one of the B best one-step lookahead values are pruned. The number of times this process takes place is equal to the size of the lookahead.

Since the number of controls that are expanded from every state at every stage is limited, the computation required to find pruned policies is not exponential in the number of platforms. However, the computation is still exponential in the size of the lookahead.

4. Platform Decomposition

We now present an approach that involves exploiting the structure of our specific problem and decomposing it into a set of simpler problems. In particular, we decompose the problem into a separate sub-problem for each platform. This sub-problem consists of determining the optimal sequence of nodes, or path, to visit assuming that platform was the only one available. The optimal solution to each sub-problem can be found analytically. After a sub-problem is solved for a particular platform and before the next sub-problem is solved, the value of each node in the associated path is updated to the value of the node multiplied by the probability that the node was not visited by the platform. This allows platforms to take into account paths assigned to previously scheduled platforms. When all of the sub-problems have been solved, a set of paths for each platform results. An outline of the platform decomposition approach is given below.

1. Assume that the platforms are ordered 1,2,...,*V*, and start with platform *i*=1.
2. Solve the single-platform problem optimally by finding a path or sequence of nodes $(n_{i1}, n_{i2}, \ldots, n_{iN})$ that the platform should attempt to visit in order to maximize its expected value (which consists of collected node values plus the reward for the platform returning to the base station if n_N is the base node).
3. For every node in the path obtained in (2), scale the value of the node to 1 minus the probability that the node will be visited by platform *i*. This allows platforms that are scheduled later to take into account the path assigned to the current platform.
4. If *i* is less than the number of platforms, then let *i*=*i*+1 and go to (2). Otherwise, we are done.

The single-platform problem in step 2 can be solved using dynamic programming or by exhaustively considering all possible paths with *N* nodes. The computation required in either case is $O(D^N)$, where *N* is the number of stages and *D* is the average degree of a node. For sparsely connected graphs, the computation required is minimal.

The set of sub-problems can be solved once for a particular ordering of platforms or multiple times for various platform orderings. We will discuss several possibilities in the next section.

The platform decomposition heuristic yields for each platform *i* a path $(n_{ij}, n_{i(j+1)}, \ldots, n_{iN})$, where *j* is the stage at which the heuristic is applied. This heuristic can be applied once before the mission begins to obtain a policy in which platform *i* attempts to visit node n_{ij} during the *j*th stage if it has not yet been destroyed. The heuristic can also be applied at every stage (for platforms that are still alive) using up-to-date state information, obtaining a policy in which platform *i* attempts to visit node n_{ij} during the *j*th stage. Finally, the heuristic can also be used to compute a value-to-go approximation for limited lookahead policies.

One of the main advantages to the platform decomposition approach is that the computation required is considerably smaller than limited lookahead policies. Assuming that the number of platform orderings considered remains fixed, the computation grows linearly in the number of platforms. In addition, as will be seen below, the method obtains solutions that are very close to the optimal. Unfortunately, while limited lookahead policies generalize easily to other problems, other problems may not have structures that easily decompose into sub-problems.

5. Computational Results

We now present some computational results from applying the above approaches to the problem described in Section 2. We consider a problem with *N*=10 stages, and either three or four platforms. The return rewards for the platforms were set to 12.7, 17.5, 19.2, and 55.0, and the most valuable platform was not included in the three-platform problems.

5.1. Limited Lookahead Policies

A limited lookahead policy consists of two main elements: the lookahead horizon, and the approximation of the value-to-go. We vary the size of the horizon from one to three and consider a number of approximations to the value-to-go. While there is some difference in the complexity of the value-to-go approximations, each one is straightforward to compute.

In many of our approaches, the value-to-go approximation for a particular state *x* after the first *k* stages, $\tilde{J}_k(x)$, involves heuristically generating for each platform *i*, a path or sequence of nodes $(n_{i(k+1)}, n_{i(k+2)}, \ldots, n_{iN})$ to attempt to visit during the remaining *N-k* stages. We denote this collection of paths *P(x,k)*. Assuming each platform attempts to visit the nodes in its path, we can determine the expected collected value resulting from visiting nodes not visited during the first *k* stages:

$$C[P(x,k)] = \sum_{\substack{\text{nodes } n \text{ not}\\ \text{yet visited}}} \left(1 - \prod_{\text{platforms } i} (1 - p_{in})\right) c_n .$$

In the above equation, c_n is the one-time value associated with node *n*, and p_{in} is the probability that platform *i* visits node *n*:

$$p_{in} = \begin{cases} \prod_{j=k+1}^{l-1} p(n_{ij}, n_{i(j+1)}), & \text{if } n_{il} = n \text{ for some } l, \\ 0, & \text{otherwise,} \end{cases}$$

where $p(n_{ij}, n_{i(j+1)})$ is the probability of successfully traversing the arc connecting nodes n_{ij} and $n_{i(j+1)}$. To understand the expression for $C[P(x,k)]$, note that the term $\prod_{\text{platforms } i}(1-p_{in})$ provides the probability that none of the platforms successfully visits node n. The term $\left(1-\prod_{\text{platforms } i}(1-p_{in})\right)c_n$ then provides the expected collected value at node n (the probability that at least one platform successfully visits the node multiplied by the node value).

We can also determine the expected reward resulting from platforms returning to the base node:

$$R[P(x,k)] = \sum_{\text{platforms } i} q_i v_i \,,$$

where

$$q_i = \begin{cases} \prod_{j=k+1}^{N-1} p(n_{ij}, n_{i(j+1)}), & \text{if } n_N \text{ is the base node,} \\ 0, & \text{otherwise,} \end{cases}$$

is the probability that platform i returns to the base node and v_i is the platform return reward.

The approximations to the value-to-go that we consider are given below. As can be seen in the descriptions, many of the approximations involve a combination of the expected collected node value, $C[P(x,k)]$, and the expected platform return reward, $R[P(x,k)]$, assuming each platform attempts to visit the nodes in the paths specified in $P(x,k)$.

1. The first approach approximates the value-to-go with zero:

   $$\tilde{J}_k(x)=0.$$

2. The second approach approximates the value-to-go with the sum of the expected collected node value and the expected platform return reward collected over a set of greedy paths:

   $$\tilde{J}_k(x)=C[P_g(x,k)]+R[P_g(x,k)].$$

   The nodes along the greedy path for platform i, $(n_{i(k+1)},\ldots,n_{iN})$, are determined as follows:

   $$n_{i(j+1)} = \arg\max_{n\in\eta(n_{ij})}\{p(n_{ij},n)\cdot c_n\},$$

   where $\eta(n_{ij})$ is the set of nodes that can be reached from node n_{ij}, and n_{ik} is the node at which platform i is located after k stages.

3. The third approach approximates the value-to-go with the expected platform return reward collected over the set of "safest" paths:

   $$\tilde{J}_k(x)=R[P_s(x,k)].$$

   The safest path is that which yields the highest probability of a platform returning successfully to its base node. These paths can be computed apriori using dynamic programming. (Essentially, the computation is equivalent to solving a set of shortest path problems.)

4. The fourth approach approximates the value-to-go with the sum of the expected collected node value and the expected platform return reward collected over the set of safest paths:

   $$\tilde{J}_k(x)=C[P_s(x,k)]+R[P_s(x,k)].$$

5. The fifth approach approximates the value-to-go with the sum of the expected collected node value and the expected platform return reward collected over the set of "most valuable" paths:

   $$\tilde{J}_k(x)=C[P_m(x,k)]+R[P_m(x,k)].$$

   The most valuable path is that which yields the highest expected total value that could be attained by a single vehicle during the remaining stages assuming none of the values at any of the nodes have yet been collected. These paths can also be computed apriori using dynamic programming.

6. The sixth approach combines (4) and (5). The value-to-go is approximated with the maximum of the values determined by those approaches.

Table 1 provides the expected optimal values for the problem illustrated in Figure 1 for a three-platform problem and a four-platform problem. We have computed these values using dynamic programming, and the computation required for the four-platform problem was approximately one week on a Sun Ultra 60 workstation. Table 1 also provides the results of applying a greedy algorithm, in which each platform selects as its next node that which maximizes its expected collected value for that stage, to one thousand sample trajectories. The performance achieved in our earlier efforts of applying rollout strategies using 20 or more Monte Carlo simulations ranged on average from 600 to 610 for the four-platform problem.

Table 1 The expected optimal values and the results of applying the greedy algorithm for the three and four platform problems.

# Platforms	Expected Optimal	Greedy
Three	574.5	475.72
Four	641.0	533.89

Tables 2 and 3 provide the values averaged over one thousand sample trajectories by applying the limited lookahead polices for lookahead sizes of one to three, using the six value-to-go approximations described above. The particular approximation approach used is given in the leftmost column. As can be seen, while the 2-stage policies generally provided results that improved significantly upon those of the 1-stage policies, those of the 3-stage policies were not substantially better and in a few cases were worse than those of the 2-stage policies. The sixth value-to-go approximation seemed to yield slightly better results than the other approximations. However, the third through sixth approximations were basically comparable. Overall, these approaches improved significantly upon the greedy algorithm and were able to obtain values close to the optimal for lookahead sizes greater than one. For lookahead sizes greater than one, these approaches were also able to obtain results slightly better than those obtained using rollout strategies with Monte Carlo simulations.

Table 2 The results of applying the limited lookahead policy to the three-platform problem.

Value-to-go Approximation	1-stage	2-stage	3-stage
1	491.09	539.58	543.40
2	520.57	543.95	553.74
3	506.55	550.82	553.76
4	500.69	529.98	559.46
5	554.10	557.94	561.75
6	555.97	563.45	561.09

Table 3 The results of applying the limited lookahaead policy to the four-platform problem.

Value-to-go Approximation	1-stage	2-stage	3-stage
1	543.32	574.24	582.75
2	589.56	607.31	593.08
3	581.48	613.29	615.63
4	574.06	618.55	619.45
5	582.84	594.09	606.96
6	595.44	615.10	624.16

Tables 4 and 5 provide the average values obtained over the same thousand sample trajectories by applying the pruned limited lookahead polices for lookahead sizes of two and three, using the value-to-go approximations described above. (Note that a pruned one-step lookahead policy is equivalent to the fully expanded one-step lookahead policy.) As can be seen, the results of these approaches do not vary significantly from the fully expanded lookahead policies. In some cases, the pruned policies performed one or two percent worse and in other cases, they performed one or two percent better.

Table 4 The results of applying the pruned limited lookahead policy to the three-platform problem.

Value-to-go Approximation	2-stage	3-stage
1	538.56	523.48
2	532.70	551.10
3	550.82	553.56
4	552.47	559.46
5	556.38	555.47
6	561.22	563.82

Table 5 The results of applying the pruned limited lookahead policy to the four-platform problem.

Value-to-go Approximation	2-stage	3-stage
1	573.19	575.21
2	605.57	607.21
3	608.98	616.21
4	613.23	615.38
5	595.55	592.50
6	613.49	617.04

5.2. Platform Decomposition Results

In applying platform decomposition to our problem, we considered the following approaches to ordering the platforms:

1. A single ordering in ascending order of the platform return reward.
2. All possible orderings.
3. A "rollout" of the ordering in (1) as described by Bertsekas, Tsitsiklis and Wu ([4]). I.e., assuming that the first *i*-1 platforms have been selected, the *i*th platform is determined as follows:
   i. Consider each remaining platform in turn as the next platform and leave the other vehicles in their original order.
   ii. Solve the set of single-platform problems in the given order.
   iii. Select as the *i*th platform that which yields the best result.

As mentioned in Section 4, there are several ways to apply the heuristic:

- The heuristic can be applied once to obtain a policy for all stages.
- The heuristic can be applied at every stage to obtain a control for the current stage using current state information.
- The heuristic can be used to generate a value-to-go approximation for a limited lookahead policy.

Table 6 provides the average values obtained over the same thousand sample trajectories by the platform decomposition approach. The result of applying the heuristic for all possible orderings and following the paths obtained for all stages is provided in the first row. The next three rows provide the results when the heuristic using the three orderings described above (least expensive to most expensive, all possible orderings, and a rollout of the orderings) is reapplied at every stage to obtain the current control. The remaining rows provide the results when the heuristic is used to provide a value-to-go approximation for a one-stage limited lookahead policy using the orderings described above is used. As can be seen, these approaches performed extremely well. The heuristic alone performed comparably to 2-stage lookahead policies, and the other variations were able to obtain strategies that yielded results that were less than one percent from the optimal expected results.

Table 6 The results of applying platform decomposition approaches. The first row provides the result of applying the heuristic for all possible platform orderings before the start of the mission and following the resulting paths. The next three rows provide the results of reapplying the heuristic at every stage using various platform orderings (1: least expensive to most expensive; 2: all possible orderings; 3: a rollout of the orderings). The last three rows provide the results of applying one-stage limited lookahead policies using the values obtained from the platform decomposition heuristic (under the various platform orderings) as an approximation to the value-to-go.

	3 platforms	4 platforms
Heuristic alone	550.85	608.89
Heuristic reapplied-1	568.81	634.97
Heuristic reapplied-2	573.83	637.81
Heuristic reapplied-3	573.83	637.81
1-stage LL-1	570.97	633.04
1-stage LL-2	571.29	635.65
1-stage LL-3	571.29	635.65

5.3 Computation Times

The following table provides the average on-line computation time (in seconds) to apply the approaches described above to one hundred sample trajectories of the four-platform problem. The off-line computation time for the limited lookahead policies was negligible. We have measured the time required to compute the controls. In practice, this time is critical since it must be within the real-time constraints of the problem. The table gives the total time to compute these controls for the ten stages. Since these times depend on the state trajectory of the system, which is random, we averaged over 100 trajectories and recorded the results in Table 7. The times for the one-stage lookahead have not been included as the time required was negligible. The experimental results were conducted on a Sun Ultra 60 workstation. As can be seen from the table, the pruned lookahead policies were significantly faster than the fully expanded lookahead policies. Considering this in combination with the fact that the performances of the two versions are comparable suggests that that pruned lookahead policies may be more useful in practice. The pruned lookahead policies were also generally much faster than the rollout algorithms using Monte Carlo simulations, whose computation times varied from 5 to over 300 seconds per sample trajectory. The decomposition approaches were extremely fast, and also provided the best results. Reapplying the decomposition heuristic at every time step appears to be the best option. However, it is not clear how easily such approaches can be applied to variations of the problem.

Table 7 Time to compute the controls for ten stages under the various approaches averaged over 100 sample trajectories of the four-platform problem. The first six lines provide the times corresponding to the fully expanded and pruned limited lookahead results given in Tables 3 and 5. The next six lines provide the times corresponding to the last six platform decomposition results given in Table 6.

	2-stage lookahead		3-stage lookahead	
	Full	Pruned	Full	Pruned
LL-1	0.77	0.12	120.4	1.8
LL-2	9.41	1.04	1358	16.4
LL-3	1.35	0.22	134.7	3.4
LL-4	6.16	0.71	716.5	9.4
LL-5	9.16	0.91	796.5	8.2
LL-6	14.85	1.73	1258	14.5
PD-Heuristic reapplied-1			0.41	
PD-Heuristic reapplied-2			9.42	

PD-Heuristic reapplied-3	1.62
PD-1-stage LL-1	51.50
PD-1-stage LL-2	396.52
PD-1-stage LL-3	138.04

6. Summary

In this paper, we have considered alternatives to using on-line simulations for approximating the value-to-go for adaptive multi-platform scheduling in a risky environment. The main limitation to using rollout algorithms with on-line simulations that was determined in our previous effort was the amount of computation required to evaluate control options at every stage. We instead considered two alternatives.

The first approach involved examining control options over a limited horizon. In our experimental results, this method produced results that were slightly better than those obtained through rollout algorithms with on-line simulations with similar computation time. Computation time was reduced significantly by introducing a pruning technique without loss in performance.

The second approach involved decomposing the problem into sub-problems associated with each platform. This method produced results that were extremely close to the optimal values and required small computation times. However, while limited lookahead methods generalize well to other problems, the decomposition method requires a suitable problem structure. Furthermore, this method may not perform well for problems with an appropriate structure if the decomposed elements require significant coordination.

7. References

[1] Alden, J.M., Smith, R.L., "Rolling Horizon Procedures in Nonhomogeneous Markov Decision Processes," *Operations Research*, V. 40, 1992.

[2] Bertsekas, D.P., Castañon, D.A., "Rollout Algorithms for Stochastic Scheduling Problems," *Journal of Heuristics*, V. 5, 1999.

[3] Bertsekas, D.P., Castañon, D.A., Curry, M.L., Logan, D., "Adaptive Multi-platform Scheduling in a Risky Environment," *1999 Proceedings from Advances in Enterprise Control Symposium*, Nov 1999.

[4] Bersekas, D.P., Tsitsiklis, J.N., Wu, C., "Rollout Algorithms for Combinatorial Optimization," *Journal of Heuristics*, V. 3, 1997.

A Game Theoretical Model for Temporal Resource Allocation in an Air Campaign[*]

Debasish Ghose[†] Jason L. Speyer[‡] and Jeff S. Shamma[§]
Mechanical and Aerospace Engineering Department
University of California at Los Angeles
Los Angeles, CA 90095
Email: <debasish+speyer+shamma>@seas.ucla.edu

Abstract

In this paper we propose a multiple resource interaction model in a game-theoretical framework as a viable approach in warfare modeling. An air raid campaign using two types of aircraft against enemy troops and air defense units is taken as the basic platform to demonstrate the key ideas of this approach. Existence of saddle point in pure strategies for the single-stage game is proved. A simplified model with linear attrition models limited by resource availability is assumed to obtain closed-form expressions for the pure strategies of the players in the single stage game. A sufficient condition for the existence of pure strategy saddle point for the multi-stage game is proposed The optimal perfect information feedback strategy is shown to be stationary under this condition. An illustrative example demonstrates the key features.

1 Introduction

In this paper we address warfare modeling as a multiple resource interaction problem, modeled in a game-theoretic framework, where two adversaries (BLUE and RED) commit their resources to an arena. In this arena each player's resource inflicts attrition on its adversary's resources through an interaction sequence decided by the spatial distribution of these resources. As in large scale warfare, the resource types of the two players may differ significantly in their capabilities and operational roles. The payoff of the game is a function of the surviving resources of the two adversaries at intermediate and terminal time points. The decision process for each player involves the resource levels to be committed by each adversary at intermediate time points. The proposed framework is general enough to encompass a large class of resource interaction problems in the context of warfare models. However, this paper presents a specific application of the general multiple resource interaction model. Mathematical details are kept at a minimum and only the outlines of the proofs of important results are given.

The specific problem addressed in this paper concerns an air campaign by the BLUE forces against RED targets (TG) located in a cluster in the RED territory. The TGs are protected by several RED air defense (AD) units located on a two dimensional game board. The BLUE forces try to destroy as many TGs as possible while trying to avoid the AD units. To do this the BLUE forces employ two types of resources. The first constitutes several SEAD (suppression of enemy air defense) units that are basically deep penetration aircraft equipped with sophisticated sensor systems that detect the presence of AD units by latching on to their emitted signals and then destroy them using anti-radiation missiles [1, 2]. The objective of using SEAD units is to create a safe corridor for bomber aircraft to penetrate enemy territory. Bombers (BMB) are the second type of BLUE resources and are used to destroy TGs (primarily) as well as ADs.

The SEAD assisted air campaign problem has a significant spatial dimension in the sense that the actual locations of the TGs and ADs determine the effectiveness of SEAD and BMB missions. The general formulation does take into account the spatial dimension too. However, in this paper, the model is greatly simplified by subsuming the spatial dimension into the attrition (or loss) functions that quantify the damage suffered by the resources due to interaction with the adversary's

[*] Supported in part by the DARPA grant N66001-99-C-8511.

[†] Visiting Researcher; On leave from the Indian Institute of Science, Bangalore, India.

[‡] Professor

[§] Professor

resources. We consider only the temporal dimension of the problem in this paper and address the problem of optimal allocation of resources by the two adversaries at each stage of a multi-stage game. However, during the actual implementation, the model may be augmented by the solution to a spatial problem of selecting an optimal route at each stage. The temporal resource allocation problem at each stage is then solved for this route.

One of the earliest warfare models that uses a temporal resource allocation paradigm is the classical air war game formulated by Berkovitz and Dresher [3]. In that paper the two adversaries (RED and BLUE) are evenly matched in terms of their resource types. The solution is sought in terms of an optimal assignment of a single resource among several tasks. Specifically, both players have several aircraft in their arsenal. These aircraft have to be assigned different roles of counter air, air defense, and ground support. The spatial dimension of the problem is suppressed completely in this model. The paper by Berkovitz and Dresher [3] was a seminal work that demonstrated the application of game theory to realistic warfare modeling. Our model differs from theirs both in terms of multiplicity of resources as well as in the mode of resource allocation.

2 Formulation of the SEAD Air Campaign

In the temporal resource allocation problem we dispense with the spatial dimensions of the overall problem and assume that the air campaign takes place on a single corridor which is defended by ADs of the RED forces from SEADs and BMBs of the BLUE forces that fly from one end of the corridor (where the SEAD and BMB stations are located) to the other end of the corridor (where the target TG is located). The determination of the corridor is actually a problem in the spatial dimension and may be posed as a risk minimization problem. However, this problem is not addressed here.

A stage in a game is defined as a single sortie in which SEADs and BMBs participate. At any given stage k of the game the BLUE forces have an available SEAD strength of S_k^s and a bomber strength of S_k^b. Similarly, the RED forces have an available air defense strength of S_k^a and a ground troop strength of S_k^g. The quantities S_k^s, S_k^b, S_k^a, and S_k^g are known to the players at the beginning of a stage. An important point to note is that these strengths may not be measured in terms of numbers (of SEAD, BMB, AD, or TG), but rather they are derived from an aggregation process that models strength as capabilities that each resource group has in terms of its mission objectives. This aspect is closely related to the spatial dimensions of the problem which determines the corridor of operation and which, in turn, defines the effectiveness of specific resources against adversary's resources through loss functions.

At any given stage k of the game, the BLUE forces partition S_k^s and S_k^b as,

$$S_k^s \;=\; u_k^s + r_k^s, \quad S_k^b = u_k^b + r_k^b \tag{1}$$

where, u_k^s and u_k^b are used by the BLUE forces in the campaign at the k-th stage and r_k^s and r_k^b are kept in reserve or "rest" for later use. Thus, the decision that the BLUE forces need to take at the beginning of each stage is how much of the SEAD and BMB strengths should be used for the campaign at that stage and how much of these strengths are to be kept in reserve.

Similarly, at a stage k of the game, the RED forces have the option of keeping some of its air defenses "hidden" (or passive) while the rest can be switched on (or made active) to track and engage SEADs and BMBs. Thus, the RED forces partition its air defense strength as,

$$S_k^a = v_k^{au} + r_k^a \tag{2}$$

where, v_k^{au} is the AD strength that is used to engage SEADs and BMBs and r_k^a is the AD strength that is kept in reserve for later use. Thus, the decision variables of BLUE forces at the beginning of stage k in the temporal resource allocation game is (u_k^s, u_k^b) and for the RED forces it is v_k^{au}.

Consider the sequence of operation of a BLUE air raid campaign and the effect the choice of the decision variables have on the outcome of the game at each stage.

Step 1: The SEADs fly along a designated corridor and engage ADs located on it. The ADs and the SEADs inflict damage on each other.

$$\begin{aligned} s_k^1 &= \text{Surviving SEAD strength} \\ &= \max\{0, u_k^s - L^s(v_k^{au}, u_k^s)\} \quad (3) \\ a_k^1 &= \text{Surviving AD strength} \\ &= \max\{0, v_k^{au} - L^{as}(v_k^{au}, u_k^s)\} \quad (4) \end{aligned}$$

where, $L^s(.,.)$ defines the damage that the SEAD strength suffers when it is confronted with one unit of AD force, and $L^{as}(.,.)$ defines the damage that the AD strength suffers in its interaction with one unit of SEAD strength.

Step 2: The BMBs now fly through and are engaged by ADs on the corridor.

$$b_k^2 \;=\; \text{Surviving BMB strength}$$

$$\begin{aligned} &= \max\{0, u_k^b - L^b(a_k^1, u_k^b)\} \\ &= \max\{0, u_k^b - L^b(\max\{0, v_k^{au} \\ &\quad -L^{as}(v_k^{au}, u_k^s)\}, u_k^b)\} \qquad (5) \\ a_k^2 &= \text{Surviving AD strength} \\ &= \max\{0, a_k^1 - L^{ab}(a_k^1, u_k^b)\} \\ &= \max\{0, \max\{0, v_k^{au} - L^{as}(v_k^{au}, u_k^s)\} \\ &\quad -L^{ab}(\max\{0, v_k^{au} - L^{as}(v_k^{au}, u_k^s)\}, u_k^b)\} \qquad (6) \end{aligned}$$

where, $L^b(.,.)$ defines the damage that the ADs inflict on the BMBs and $L^{ab}(.,.)$ defines the damage that the BMBs inflict on the ADs.

Step 3: Finally the BMBs engage the TGs at the end of the corridor.

$$\begin{aligned} g_k^3 &= \text{Surviving ground troop strength} \\ &= \max\{0, S_k^g - L^g(b_k^2, S_k^g)\} \\ &= \max\{0, S_k^g - L^g(\max\{0, u_k^b \\ &- L^b(\max\{0, v_k^{au} - L^{as}(v_k^{au}, u_k^s)\}, u_k^b)\}, S_k^g)\} \qquad (7) \end{aligned}$$

where, $L^g(.,.)$ defines the damage that BMBs inflict on the TGs.

At the next stage $k+1$ the two players have the following force strengths available:

$$\begin{aligned} S_{k+1}^s &= r_k^s + s_k^1, \quad S_{k+1}^b = r_k^b + b_k^2 \\ S_{k+1}^a &= r_k^a + a_k^2, \quad S_{k+1}^g = g_k^3 \qquad (8) \end{aligned}$$

A resource interaction tableau that summarizes the above is shown in Figure 1. The complete state equations corresponding to the above system of resource interaction are,

$$\begin{aligned} S_{k+1}^s &= \max\{0, u_k^s - L^s(v_k^{au}, u_k^s)\} + (S_k^s - u_k^s) \qquad (9) \\ S_{k+1}^b &= \max\{0, u_k^b - L^b(\max\{0, v_k^{au} \\ &\quad -L^{as}(v_k^{au}, u_k^s)\}, u_k^b)\} + (S_k^b - u_k^b) \qquad (10) \\ S_{k+1}^a &= \max\{0, \max\{0, v_k^{au} - L^{as}(v_k^{au}, u_k^s)\} \\ &\quad -L^{ab}(\max\{0, v_k^{au} - L^{as}(v_k^{au}, u_k^s)\}, u_k^b)\} \\ &\quad +(S_k^a - v_k^{au}) \qquad (11) \\ S_{k+1}^g &= \max\{0, S_k^g - L^g(\max\{0, u_k^b - L^b(\max\{0, v_k^{au} \\ &\quad -L^{as}(v_k^{au}, u_k^s)\}, u_k^b)\}, S_k^g)\} \qquad (12) \end{aligned}$$

with the controls of the two players as $u_k^s \in [0, S_k^s]$, $u_k^b \in [0, S_k^b]$, $v_k^{au} \in [0, S_k^a]$. Briefly, we write the state equations as,

$$S_{k+1} = f(S_k, u_k, v_k) \qquad (13)$$

where, $S_k = (S_k^s, S_k^b, S_k^a, S_k^g)$ and $f(.,.,.)$ represents the state transitions.

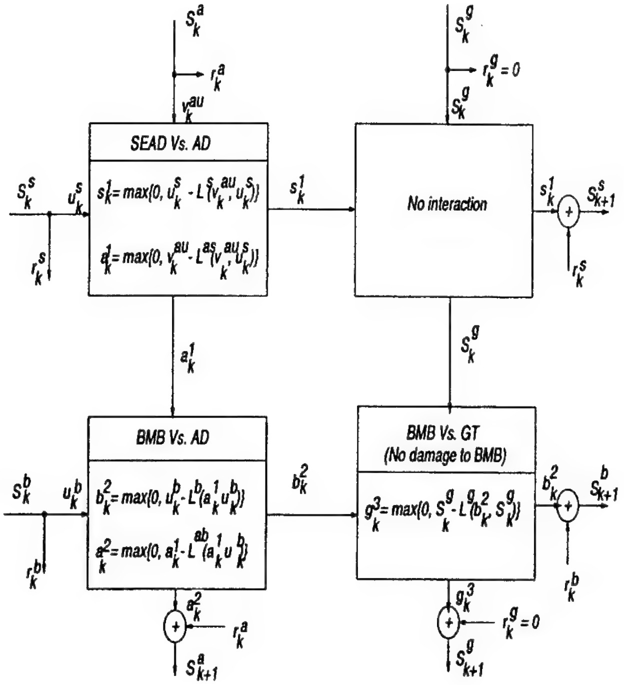


Figure 1: The resource interaction tableau for stage k for SEAD assisted air campaign

In this game we define the payoff to be the cumulative damage caused by the surviving TG strength at each stage. This would be a monotonically increasing function of the sum of the surviving TG strengths at each stage. This is the payoff that the RED forces try to maximize and the BLUE forces try to minimize. A justification of this performance criterion can be given from the viewpoint that the TGs could be of the type whose effectiveness is linked to the length of time over which it survives or is operational. The payoff at the end of the designated n stages is,

$$J = \sum_{k=1}^{n} g_k^3 \qquad (14)$$

3 Linear Attrition Model

We consider a simplified model where the potential losses or attrition are assumed to be linear functions of the adversary's resource strength restricted by the resource availability. Let,

$$\begin{aligned} L^s(v_k^{au}, u_k^s) &= \alpha v_k^{au}, \quad L^{as}(v_k^{au}, u_k^s) = \beta u_k^s \\ L^b(a_k^1, u_k^b) &= \gamma a_k^1, \quad L^{ab}(a_k^1, u_k^b) = \eta u_k^b, \\ L^g(b_k^2, S_k^g) &= \theta b_k^2 \qquad (15) \end{aligned}$$

where, α, β, γ, η, and θ are non-negative scalars. The first equation means that α SEAD strength is destroyed by one unit of AD strength. The other loss parameters have a similar interpretation. With this simplified linear model the corresponding variables at each step in a stage k are given by,

$$\begin{aligned}
s_k^1 &= \max\{0, u_k^s - \alpha v_k^{au}\} && (16)\\
a_k^1 &= \max\{0, v_k^{au} - \beta u_k^s\} && (17)\\
b_k^2 &= \max\{0, u_k^b - \gamma a_k^1\}\\
&= \max\{0, u_k^b - \gamma \max\{0, v_k^{au} - \beta u_k^s\}\} && (18)\\
a_k^2 &= \max\{0, a_k^1 - \eta u_k^b\}\\
&= \max\{0, \max\{0, v_k^{au} - \beta u_k^s\} - \eta u_k^b\} && (19)\\
g_k^3 &= \max\{0, S_k^g - \theta b_k^2\}\\
&= \max\{0, S_k^g - \theta \max\{0, u_k^b\\
&\quad -\gamma \max\{0, v_k^{au} - \beta u_k^s\}\}\} && (20)
\end{aligned}$$

The first step in solving the multi-stage problem, as posed here, is the solution of a single stage game problem. Consider the payoff at the k-th stage,

$$J_k(u_k, v_k) = S_{k+1}^g = g_k^3 =$$
$$\max\{0, S_k^g - \theta \max\{0, \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta u_k^s)\}\}\} \quad (21)$$

Suppose we want to solve the problem at the k-th stage treating it as a single stage game. That is,

$$\min_{(u_k^s, u_k^b) \in [0, S_k^s] \times [0, S_k^b]} \quad \max_{v_k^{au} \in [0, S_k^a]}$$
$$\max\{0, S_k^g - \theta \max\{0, \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta u_k^s)\}\}\} \quad (22)$$

Lemma 1 *The function $J_k(u_k, v_k)$ is a monotonically decreasing function of each component of $u_k = (u_k^s, u_k^b)$ and a monotonically increasing function of $v_k = v_k^{au}$.*

Proof. A function $f(.)$ is a monotonically decreasing function of its argument if $f(x) \le f(y)$ for every $x > y$. Similarly, it is a monotonically increasing function of its argument if $f(x) \ge f(y)$ for every $x > y$. Consider $\hat{u}_k^s > u_k^s$. Then,

$$\begin{aligned}
& u_k^b - \gamma(v_k^{au} - \beta \hat{u}_k^s) > u_k^b - \gamma(v_k^{au} - \beta u_k^s)\\
\Rightarrow\ & \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta \hat{u}_k^s)\}\\
& \ge \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta u_k^s)\}\\
\Rightarrow\ & \theta \max\{0, \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta \hat{u}_k^s)\}\}\\
& \ge \theta \max\{0, \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta u_k^s)\}\}\\
\Rightarrow\ & \max\{0, S_k^g - \theta \max\{0, \min\{u_k^b, u_k^b\\
& \quad - \gamma(v_k^{au} - \beta \hat{u}_k^s)\}\}\}\\
& \le \max\{0, S_k^g - \theta \max\{0, \min\{u_k^b, u_k^b\\
& \quad - \gamma(v_k^{au} - \beta u_k^s)\}\}\}\\
\Rightarrow\ & J_k((\hat{u}_k^s, u_k^b), v_k^{au}) \le J_k((u_k^s, u_k^b), v_k^{au})
\end{aligned}$$

Similarly, consider $\hat{u}_k^b > u_k^b$. Let $X = \gamma(v_k^{au} - \beta \hat{u}_k^s)$. Then,

$$\begin{aligned}
& \hat{u}_k^b - X > u_k^b - X\\
\Rightarrow\ & \min\{\hat{u}_k^b, \hat{u}_k^b - X\} > \min\{u_k^b, u_k^b - X\}\\
\Rightarrow\ & \theta \max\{0, \min\{\hat{u}_k^b, \hat{u}_k^b - X\}\}\\
& \ge \theta \max\{0, \min\{u_k^b, u_k^b - X\}\}\\
\Rightarrow\ & \max\{0, S_k^g - \theta \max\{0, \min\{\hat{u}_k^b, \hat{u}_k^b - X\}\}\}\\
& \le \max\{0, S_k^g - \theta \max\{0, \min\{u_k^b, u_k^b - X\}\}\}\\
\Rightarrow\ & J_k((u_k^s, \hat{u}_k^b), v_k^{au}) \le J_k((u_k^s, u_k^b), v_k^{au})
\end{aligned}$$

Finally, let $\hat{v}_k^{au} > v_k^{au}$. Then,

$$\begin{aligned}
& u_k^b - \gamma(\hat{v}_k^{au} - \beta u_k^s) < u_k^b - \gamma(v_k^{au} - \beta u_k^s)\\
\Rightarrow\ & \min\{u_k^b, u_k^b - \gamma(\hat{v}_k^{au} - \beta u_k^s)\}\\
& \le \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta u_k^s)\}\\
\Rightarrow\ & \theta \max\{0, \min\{u_k^b, u_k^b - \gamma(\hat{v}_k^{au} - \beta u_k^s)\}\}\\
& \le \theta \max\{0, \min\{u_k^b, u_k^b - \gamma(v_k^{au} - \beta u_k^s)\}\}\\
\Rightarrow\ & \max\{0, S_k^g - \theta \max\{0, \min\{u_k^b, u_k^b\\
& \quad - \gamma(\hat{v}_k^{au} - \beta u_k^s)\}\}\}\\
& \ge \max\{0, S_k^g - \theta \max\{0, \min\{u_k^b, u_k^b\\
& \quad - \gamma(v_k^{au} - \beta u_k^s)\}\}\}\\
\Rightarrow\ & J_k((u_k^s, u_k^b), \hat{v}_k^{au}) \ge J_k((u_k^s, u_k^b), v_k^{au})
\end{aligned}$$

This completes the proof of the monotonicity property of the payoff function at the k-th stage when the game is treated as a single stage game. □

Below we state the fundamental minimax theorem by Fan [5] which will be used to prove the existence of saddle points in pure strategies.

Theorem 1 Fan's minimax theorem. *Let X, Y be two compact Hausdorff spaces and f a real-valued function defined on $X \times Y$. Suppose that, for every $y \in Y$, $f(x, y)$ is lsc on X; and for every $x \in X$, $f(x, y)$ is usc on Y. Then the equality $\min_{x \in X} \max_{y \in Y} f(x, y) = \max_{y \in Y} \min_{x \in X} f(x, y)$ holds if and only if for any two finite sets $\{x_1, x_2, \ldots, x_n\} \subset X$ and $\{y_1, y_2, \ldots, y_m\} \subset Y$, there exist $x_0 \in X$ and $y_0 \in Y$ such that $f(x_0, y_i) \le f(x_j, y_0)$ for all $1 \le j \le n$ and $1 \le i \le m$.*

Theorem 2 *A saddle point in pure strategies exists for the k-th stage of the game with performance index given in (21).*

Proof. Since J_k is jointly continuous with respect to u_k and v_k and the control sets are intervals on the real line (and therefore compact), by standard results in game theory [4], the game admits a saddle point in mixed strategies. Since the payoff function does not satisfy the convexity-concavity property normally used for proving existence of pure strategies, we invoke the fundamental theorem by Fan [5] to prove the existence of pure strategies. Define $\hat{u}_k = (S^s_k, S^b_k)$ and $\hat{v}_k = (S^a_k)$. Then, from the monotonicity of J_k with respect to u_k and v_k, for any $u_k \in [0, S^s_k] \times [0, S^b_k]$ and $v_k \in [0, S^a_k]$, we have $J_k(\hat{u}_k, v_k) \leq J_k(u_k, \hat{v}_k)$. Since this result is true for any u_k and v_k, it is also true for any finite sets of u_k's and v_k's selected from $[0, S^s_k] \times [0, S^b_k]$ and $[0, S^a_k]$, respectively. Then, by Fan's minimax theorem the game has a saddle point in pure strategies. $\Box$

In fact, in the above, $(\hat{u}_k, \hat{v}_k)$ itself is a saddle point. However, this saddle point may not be unique and there could exist multiple saddle points. It turns out that for this simplified model the optimal strategies for the players at the k-th stage, when solved as a single stage game, can be obtained in the closed-form as,

If $(S^a_k) \in \mathcal{M}$, then

$$v^{au*}_k = S^a_k \tag{23}$$

If $(S^a_k) \notin \mathcal{M}$, then

$$v^{au*}_k \quad \in \quad \{[0, S^a_k] \backslash \mathcal{M}\} \tag{24}$$

where, '\' denotes the set difference operation.

If $(S^s_k, S^b_k) \in \mathcal{N}$, then

$$u^{b*}_k = S^b_k \tag{25}$$

and

$$u^{s*}_k = S^s_k \quad \text{if} \quad S^a_k/\beta > S^s_k \tag{26}$$

$$u^{s*}_k \in [S^a_k/\beta, S^s_k] \quad \text{otherwise} \tag{27}$$

If $(S^s_k, S^b_k) \notin \mathcal{N}$, then

$$(u^{s*}_k, u^{b*}_k) \quad \in \quad \{\{[0, S^s_k] \times [0, S^b_k]\} \backslash \mathcal{N}\} \tag{28}$$

The sets $\mathcal{N}$ and $\mathcal{M}$ are defined as,

$$\mathcal{N}_1 \;=\; \left\{ (x^{us}, x^{ub}) \;:\; x^{us} \geq \frac{S^a_k}{\beta};\; x^{ub} < \frac{S^g_k}{\theta} \right\} \tag{29}$$

$$\begin{aligned} \mathcal{N}_2 \;=\; & \left\{ (x^{us}, x^{ub}) \;:\; x^{us} < \frac{S^a_k}{\beta}; \right. \\ & \left. x^{ub} < \gamma(S^a_k - \beta x^{us}) + \frac{S^g_k}{\theta} \right\} \end{aligned} \tag{30}$$

$$\mathcal{N} \;=\; \mathcal{N}_1 \cup \mathcal{N}_2 \tag{31}$$

$$\mathcal{M} \;=\; \{y^{va} : y^{va} < S^b_k/\gamma + \beta S^s_k\} \tag{32}$$

where, x^{us}, x^{ub}, and y^{va} are variables that correspond to the SEAD, BMB, and AD resource strengths, respectively. The sets $\mathcal{M}$ and $\mathcal{N}$ are such that the BLUE player will not be able to destroy RED TGs completely if it confines its allocation to the set $\mathcal{N}$. Similarly, the RED player will not be able to protect its TGs completely (so that they remain undamaged) if it confines its allocation to the set $\mathcal{M}$. A schematic representation of the optimal allocation for BLUE is given in Figure 2. The optimal allocations are in the shaded region shown in the figure if the available resource strengths are such that the point (S^s_k, S^b_k) does not lie in the interior of $\mathcal{N}$. In which case, any allocation in the shaded region is optimal and will destroy the TGs completely. We will call these solutions as "non-dominated" and denote them as ND. Otherwise, if the point lies in the interior of $\mathcal{N}$, then (S^s_k, S^b_k) is the optimal allocation. These solutions are called "dominated" and denoted as D. Similarly, if S^a_k lies in the interior of $\mathcal{M}$ then S^a_k is the optimal allocation, and is a "dominated" solution. Otherwise, the optimal allocation would be any point in $[S^b_k/\gamma + \beta S^s_k, S^a_k]$ and is called "non-dominated". Each such ND allocation would destroy the BLUE BMBs completely so that no damage would be inflicted on the TGs. From the fact that the interactions occur in a well-defined sequence and attritions to BMBs and GTs occur in separate interaction blocks, it can be shown that the players cannot both have ND solutions at any given stage.

Although, depending on the available resource levels, the game admits multiple saddle points in pure strategies, it is logical for the players to avoid using excessive resources. This implies that the RED forces will use,

$$v^{au*}_k = \min\left\{S^b_k/\gamma + \beta S^s_k, S^a_k\right\} \tag{33}$$

and the BLUE forces will select a Pareto point from its solution set given in (28). The Pareto set is shown in Figure 2 as the bold line when the available resources are not in the interior of $\mathcal{N}$. The optimal allocation would be the available resources of BLUE when the available resources are in the interior of $\mathcal{N}$. Whenever a ND solution exists for BLUE at any given stage, it has the capability to destroy all the TGs in that stage. Similarly, if a ND solution exists for RED then it has the capability to destroy all the BMB resource used by BLUE in that stage.

Now, consider the multi-stage game. The payoff kernel of the game at the stage k is defined as,

$$J_k(S_k, u_k, v_k) + V_{k+1}(f(S_k, u_k, v_k)) \tag{34}$$

where, $V_k(S_k)$ is the value of the game at stage k, obtained when players play optimally. The optimal payoff is given by,

$$\begin{aligned} V_k(S_k) \;=\; & \min_{u_k} \max_{v_k} [J_k(S_k, u_k, v_k) \\ & + V_{k+1}(f(S_k, u_k, v_k))] \end{aligned} \tag{35}$$

If a saddle point exists then the solution of the above problem gives the optimal strategies of the players at

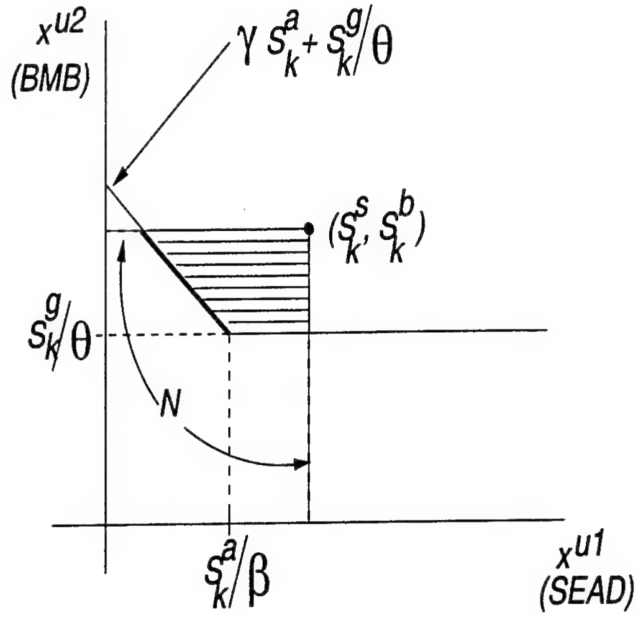


Figure 2: Optimal resource allocation and the Pareto minimum set for BLUE

the k-th stage. The optimal payoff of the game is given by $V_0(x_0)$.

Theorem 3 *If $J_k(S_k, u_k, v_k) + V_{k+1}(f(S_k, u_k, v_k))$ is a monotonically decreasing function of u_k and a monotonically increasing function of v_k for all k, then the multi-stage game has a saddle point in pure strategies at each stage k.*

Proof. The proof uses similar arguments as in Theorem 2 which depends only on the monotonicity property of the objective function to invoke Fan's theorem. ▯

The condition stated in Theorem 3 is quite restrictive and is a sufficient condition for the existence of saddle points. However, it is not necessary. Fan's theorem does not require monotonicity to be satisfied. The monotonicity conditions are actually themselves sufficient conditions for Fan's theorem to hold. In fact, we can relax the above condition further as follows,

Theorem 4 *If $J_k(S_k, u_k, v_k) + V_{k+1}(f(S_k, u_k, v_k))$ satisfies the property that*

$$\begin{aligned} & J_k(S_k, (S_k^s, S_k^b), v_k^{au}) + V_{k+1}(f(S_k, (S_k^s, u_k^b), v_k^{au})) \\ & \quad \leq \quad J_k(S_k, (u_k^s, u_k^b), v_k^{au}) + V_{k+1}(f(S_k, (u_k^s, u_k^b), v_k^{au})), \\ & \qquad\qquad \text{for all } u_k \text{ and a fixed } v_k \\ & J_k(S_k, (u_k^s, u_k^b), S_k^a) + V_{k+1}(f(S_k, (u_k^s, u_k^b), S_k^a)) \\ & \quad \geq \quad J_k(S_k, (u_k^s, u_k^b), v_k^{au}) + V_{k+1}(f(S_k, (u_k^s, u_k^b), v_k^{au})), \\ & \qquad\qquad \text{for all } v_k^{au} \text{ and a fixed } u_k \end{aligned}$$

for all k then the multi-stage game has a saddle point in pure strategies at each stage k.

Proof. The proof uses similar arguments as in Theorem 2. The conditions given above ensures that for each player there exists a choice of resource levels that satisfy the conditions of Fan's theorem. ▯

The fact that monotonicity is not required is seen in the example in Section 4 where Theorem 4 is satisfied but not Theorem 3.

If the optimal pure strategies for the players at each stage are $u_1^*, \ldots, u_N^*$ and $v_1^*, \ldots, v_N^*$, then we may construct the optimal pure strategies for the multi-stage game as $u^* = (u_1^*, \ldots, u_N^*)$ and $v^* = (v_1^*, \ldots, v_N^*)$. We show this in the following theorem.

Theorem 5 *If the conditions given in Theorem 3 or Theorem 4 holds then a saddle point pure strategy for the multi-stage game is given by the optimal solution of the single-stage game at each stage.*

Proof. Suppose at a given stage k both players have only D solutions then any deviation from the single stage saddle point solution would result in higher surviving resource strengths of the other player. The surviving TG strength will accordingly decrease or increase. If BLUE has a ND solution in a given stage and deviates from it (that is, uses a D solution), then the payoff in that stage is non-zero, thus increasing the total payoff. Similarly, if RED has a ND solution and deviates from it in that stage (and uses a D solution), then the surviving TG strength decreases thus reducing the payoff in that stage. In subsequent stages the payoff is either positive or zero and so the deviation by RED increases the payoff. These observations are adequate to prove that the single-stage saddle point solution is a stationary saddle point solution for the multi-stage game if the conditions in Theorem 3 or 4 hold. ▯

The monotonicity conditions in Theorem 3 and those in Theorem 4 are somewhat stringent and are not easy to verify for a game with a large number of stages. However, for games with smaller number of stages it might be possible to verify this condition computationally. If neither of these conditions hold then the optimal strategies of the players are likely to be mixed behavioral strategies. Even if they are pure strategies they may no longer be stationary. Below we will solve an example where the specified conditions are indeed met and we obtain optimal pure strategies that are stationary.

Although the above results are obtained for a linear loss function model limited by resource availability,

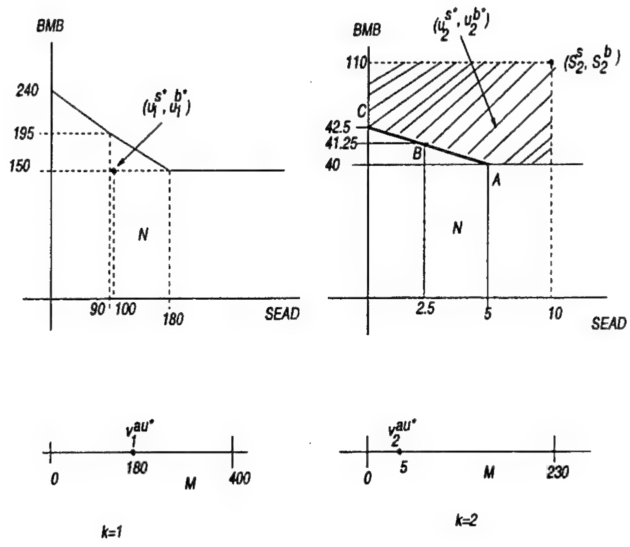


Figure 3: Optimal SEAD and BMB allocation and the Pareto minimum set for BLUE in Stages 1 and 2

since only the monotonicity property of the payoff function and surviving resource levels are used, it should be possible to extend these results to nonlinear loss functions that themselves satisfy suitable monotonicity properties. However, closed-form solutions may no longer be possible and optimal solutions have to be computationally obtained.

4 Illustrative Examples

To illustrate the utility of the game theoretical framework for warfare modeling we present an example here. Let the initial resource strengths for SEAD be $S_1^s = 100$, for BMB be $S_1^b = 150$, for AD be $S_1^a = 180$, and for TG be $S_1^g = 300$. Let the attrition coefficients be $\alpha = 0.5$, $\beta = 1$, $\gamma = 0.5$, $\eta = 0.5$, $\theta = 2$. The sets $\mathcal{N}$ and $\mathcal{M}$ will be as shown in Figure 3. We see that in stage $k = 1$ both the optimal solutions are of type D. Assuming that the condition in Theorem 3 holds (we will verify this later), the optimal allocations should be,

$$v_1^{au*} = 180, \quad u_1^{s*} = 100, \quad u_1^{b*} = 150$$

With this allocation the surviving resources at the end of Stage 1 are,

$$S_2^s = 10, \quad S_2^b = 110, \quad S_2^a = 5, \quad S_2^g = 80$$

But in Stage 2, the computation of $\mathcal{N}$ and $\mathcal{M}$ shows that BLUE now has a ND solution while RED has a D solution. Further, BLUE has multiple Pareto solutions, any of which could be used to destroy the TGs completely. Note that BLUE now has the option of a trade-off between its two resources of SEADs and BMBs. These Pareto solutions may be parmeterized as $(u_2^{s*}, u_2^{b*}) = \rho(0, 42.5) + (1 - \rho)(5, 40)$ with respect to a parameter $\rho \in [0, 1]$. Any one of these is sufficient to destroy the TGs completely. The optimal solution for RED is to use $v_2^{au*} = 5$. These solutions are shown in Figure 3. The game thus ends after 2 stages with a total payoff of 80.

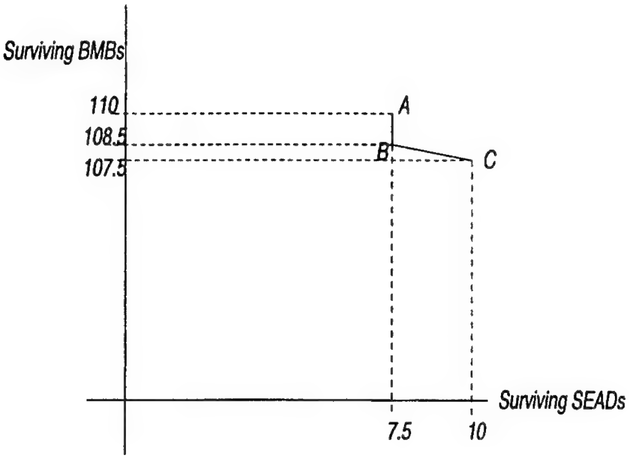


Figure 4: Surviving SEADs and BMBs after Stage 2

A logical question now would be how would BLUE select among the available multiple Pareto solutions in Stage 2. A possible approach to this problem could be to examine the surviving resources of the players. It turns out that when BLUE uses its optimal solutions all of the RED resources are destroyed at the end of Stage 2. However, the surviving BMB and SEAD resource levels are as shown in Figure 4 where the points A, B, and C correspond to the points in Figure 3. Obviously, A is a better choice than B, but one could again perform a trade-off between choices A (for which 7.5 SEADs and 110 BMBs survive) and the choices (B, C] (for which the surviving SEADs and BMBs are in ((7.5, 108.75), (10, 107.5)] depending on the relative value of SEAD strength and BMB strength.

Let us consider a few other possible strategies and examine how these compare with the optimal strategy given here. Let ν_k denote the fraction of its resources (both SEAD and BMB) that the BLUE forces deploy at the k-th stage. Let the strategy followed by BLUE be such that ν_1 takes values between 0 and 1 while $\nu_k = 1$ for $k \geq 2$. Note that $\nu_1 = 1$ corresponds to the optimal strategy. The results are as given in table 1. The variable S_k^g denotes the surviving TG strength after the stage $k - 1$.

To verify that the conditions stated in Theorem 4

Table 1: Payoffs for various strategies

ν_1	payoff	S_2^g	S_3^g	S_4^g
1	80	80	0	0
0.9	120	120	0	0
0.75	188.75	180	8.75	0
0.5	432.5	280	152.5	0

hold in this game we need to show that

$$\begin{aligned} & J_1((u_1^s, u_1^b), v_1^{au}) + J_2((S_2^s, S_2^b, S_2^a) \\ = \; & J_1((u_1^s, u_1^b), v_1^{au}) \\ & +J_2((\max\{0, u_k^s - \alpha v_k^{au}\} + S^s - u_1^s, \\ & \max\{0, u_k^b - \gamma \max\{0, v_k^{au} - \beta u_k^s\}\} + S^b - u_1^b, \\ & \max\{0, \max\{0, v_k^{au} - \beta u_k^s\} - \eta u_k^b\}\} + S^a - v_1^{au}) \end{aligned}$$

satisfies the property that it attains (i) its minimum with respect to (u_1^s, u_1^b) at $u_1^s = S_1^s$ and $u_1^b = S_1^b$, while v_1^{au} is held constant (ii) its maximum with respect to v_1^{au} at $v_1^{au} = S_1^a$, while u_1^s and u_1^b are held constant. This can be verified computationally. It can also be verified that this game does not satisfy the monotonicity condition as stated in Theorem 3. We omit details.

Finally, we present another example in which the conditions of Theorems 3 and 4 do not hold and the game does not have a pure strategy saddle point. It also does not have a stationary strategy. Consider a similar game in which the SEADs do not have a role to play. Which means that $\alpha = 0$ and $\beta = 0$. The other parameters are, $\gamma = 5$, $\eta = 1$, and $\theta = 1$. This corresponds to a scenario where there are only BMBs that interact with ADs and each unit of AD strength can destroy 5 units of BMBs. On the other hand each unit of BMBs can destroy just one unit of AD strength and one unit of TG. The initial conditions are $S_1^b = 30$, $S_1^a = 5$ and $S_1^g = 100$. We solve this game for two stages. The single stage optimality conditions can be directly used to obtain an expression for the optimal payoff P if BLUE uses u_1 BMBs and RED uses v_1 ADs in the first stage, and then both play optimally in the second stage.

$$\begin{aligned} P = & 200 - 2\max\{0, u_1 - 5v_1\} - \max\{0, 5 - (u_1 - 5v_1) \\ + & \max\{0, u_1 - 5v_1\} - 5\max\{0, v_1 - u_1\}\} \qquad (36) \end{aligned}$$

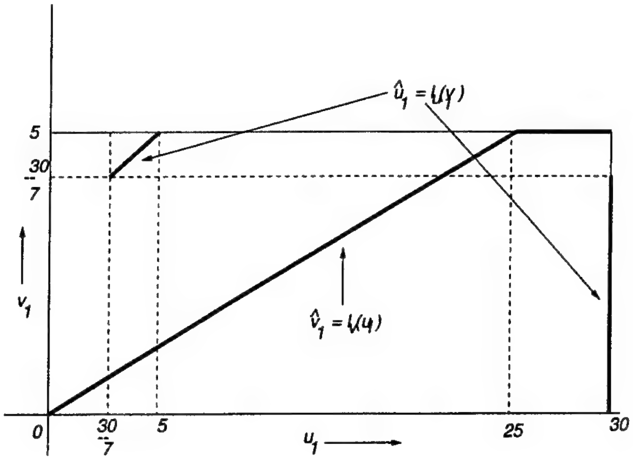


Figure 5: The rational reaction curves

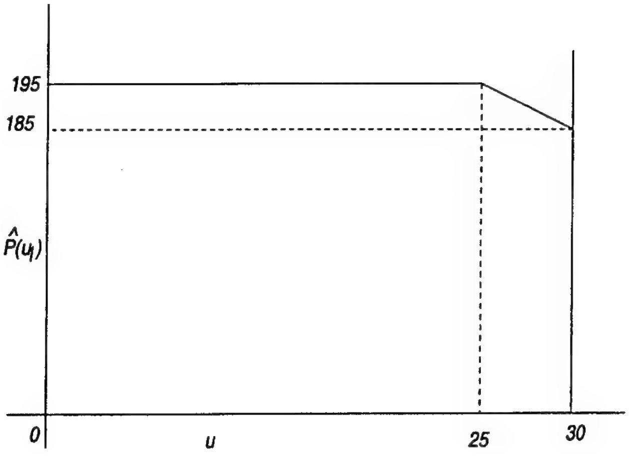


Figure 6: Optimal payoff to BLUE

The payoff function is continuous and so, for every choice of $u_1 \in [0, S_1^b]$, there exists an optimal choice $\hat{v}_1 = l_v(u_1)$ of RED which maximizes the payoff. Here $l_v(.) : [0, S_1^b] \to [0, S_1^a]$ denotes the rational reaction function of RED. Also, the maximum payoff is denoted by $\hat{P}(u_1)$. Similarly, for every choice of $v_1 \in [0, S_1^a]$, there exists an optimal choice $\hat{u}_1 = l_u(v_1)$ of BLUE that minimizes the payoff. Here $l_u(.) : [0, S_1^a] \to [0, S_1^b]$ denotes the rational reaction function of BLUE. This maximum payoff is denoted by $\hat{P}(v_1)$. Plotting these quantities in Figures 5-7, we observe the following:

(i) There is a discontinuity in the rational reaction curve of BLUE.

(ii) If we consider only pure strategies for the players then minimax value of the game (=185) is not equal to the maximin value (=$177\frac{6}{7}$).

All this implies that the 2 stage game does not have a pure strategy saddle point. Figure 5 also shows that the conditions mentioned in Theorems 3 and 4 are both violated in this example. Also, it is obviously not possible for BLUE to have an optimal pure strategy since if it does, then the optimal reaction to it would be a pure strategy for RED.

Suppose we assume that RED has a pure strategy. The only possibility seems to be $v_1^* = 30/7$ since any

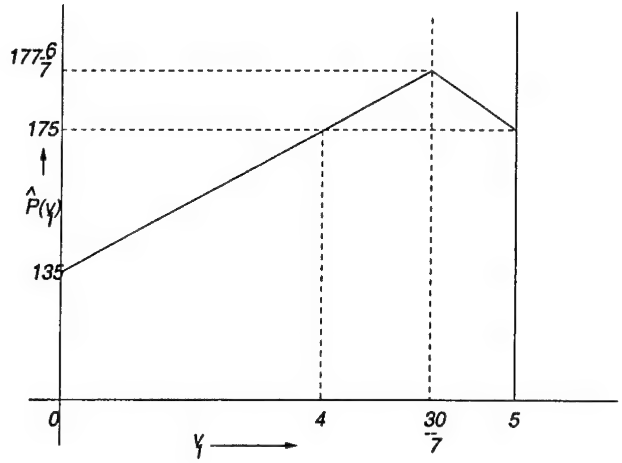


Figure 7: Optimal payoff to RED

Figure 8: Payoff for $v_1^* = 30/7$

other choice would imply an optimal pure strategy reaction from BLUE. Plotting the payoff for this choice of RED against all possible pure strategy choices of BLUE yields Figure 8. The figure shows that u^*_{1a} and u^*_{1b} are the possible supports for the mixed strategy reaction of BLUE. If it were true then we should have, for some $p \in [0, 1]$,

$$\arg\max_{v_1}\{pP(30/7, v_1) + (1-p)P(30, v_1)\} = 30/7 \quad (37)$$

To see if such a p exists or not we plot the payoff against v_1 for various values of p in Figure 9. It can be easily seen that for no value of p does this curve attain its maximum at $v = 30/7$. This implies that either (i) RED does not have an optimal pure strategy and so we must look for a mixed strategy for RED too, or (ii) The support of BLUE's mixed strategy is a larger set than just u^*_{1a} and u^*_{1b}, or (iii) Both of the above.

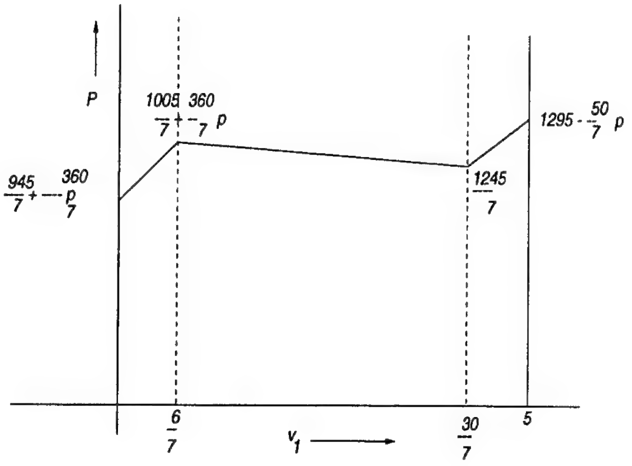


Figure 9: General trend of the payoff against p

This example, although simple, shows that in certain cases when the conditions of Theorem 3 and 4 are not met, the solution of the game must be sought in terms of mixed strategies. Also, these mixed strategies may not be easily computable in terms of a finite support, that is, as a probability distribution on a finite set of pure strategies. It also shows that even simple resource interaction problems can give rise to a rich strategy space. Computational algorithms are presently under development to solve this class of problems with a larger number of stages, different performance criteria, non-linear attrition functions, and interaction between multiple resources.

5 Conclusions

A game theoretical framework for a war game involving an air campaign against an adversary's target resource protected by air defense units is proposed and modeled as a multiple resource interaction problem. Focusing only on the temporal aspect of the game, existence of optimal pure strategies to allocate the resources of the two adversaries is proved under certain conditions. Closed-form solutions are also obtained for attrition functions that are linear within the bounds of resource availability. An illustrative example is worked out to demonstrate the ideas presented in the paper. The approach shows the strong potential that game theoretical concepts have on planning campaigns from a higher level command point of view. Further work in this direction involves the computation of non-stationary pure and mixed strategy when they exist, the extension of the model to its spatial dimension, incorporating multiple interactions among resources to account for approximations introduced due to aggregation of interaction events, development of computational algorithms to compute optimal strategies for large-scale interactions, and incorporation of non-linear attrition functions.

References

[1] P.K. Davis and J.H. Bigelow (1998): *Experiments in Multi-Resolution Modeling (MRM)*, RAND Publication, MR-1004-DARPA, 1998.

[2] D. Vaughan *et al.* (1998): *Capturing the Essential Factors in Reconnaissance and Surveillance Force Sizing and Mix*, Project Air Force, RAND documented Briefing, DB 199-AF, 1998.

[3] L.D. Berkovitz and M. Dresher (1959): A game-theory analysis of tactical air war, *Operations Research*, vol. 7, 1959, pp. 599-620.

[4] T. Basar and G.J. Olsder (1995): *Dynamic Noncooperative Game Theory*, 2nd Edition, Academic Press, London, 1995.

[5] K. Fan (1953): Minimax theorems, *Proceedings of the National Academy of Sciences*, Vol. 39, pp. 42-47.

DECEPTION IN NON-COOPERATIVE GAMES WITH PARTIAL INFORMATION[†]

João P. Hespanha[‡] hespanha@usc.edu

Yusuf S. Ateşkan ateskan@usc.edu

Hüseyin H. Kızılocak kiziloca@usc.edu

Dept. of Electrical Engineering—Systems, Univ. of Southern California
3740 McClintock Ave., Room 318, MC 2563, Los Angeles, CA 90089-2563
phone: (213) 740-9137, fax: (213) 821-1109

Abstract

In this paper we explore how deception can be used by rational players in the context of non-cooperative stochastic games with partial information. We show that, when one of the players can manipulate the information available to its opponents, deception can be used to increase the player's payoff by effectively rendering the information available to its opponent useless. However, this is not always the case. Paradoxically, when the degree of possible manipulation is high, deception becomes useless against an intelligent opponent since it will simply ignore the information that has potentially been manipulated. This study is carried out for a prototype problem that arises in the control of military operations, but the ideas presented are useful in other areas of applications, such as price negotiation, multi-object auctioning, pursuit-evasion, etc.

1 Introduction

Competitive games are usually classified as either having full or partial information. In *full-information* games both players know the whole state of the game when they have to make decisions. By state, we mean all information that is needed to completely describe the future evolution of the game, when the decision rules used by both players are known. Examples of full information games include Chess, Checkers, and Go. *Partial-information* games differ from these in that at least one of the players does not know the whole state of the game. Poker, Bridge, and Hearts are examples of such games. In full information games, as a player is planning its next move, it only needs to hypothesize over its and the opponent's future moves to predict the possible outcomes of the game [1]. This is key to using dynamic programming to solve full-information games. Partial information games are especially challenging because this reasoning may fail. In many partial information games, to predict the possible outcomes of the game, a player must hypothesize not only on the future moves of both players, but also on the past moves of the opponent. This often leads to a tremendous increase in the complexity of the games. In general, partial information stochastic games are poorly understood and the literature is relatively sparse. Notable exceptions are games with lack of information for one of the players [2, 3] and games with particular structures such as the Duel game [4], the Rabbit and Hunter game [5], the Searchlight game [6, 7], etc.

Another issue that makes partial information games particularly interesting is the fact that a player can obtain future rewards by either one of two possible mechanisms:

1. Choosing an action that will take the game to a more favorable state;
2. Choosing an action that will make the other player act in our own advantage by making it believe that the game is in a state other than the actual one.

The latter corresponds to a deception move and is only possible in partial information games.

[†] This research was supported by the Defense Advanced Research Projects Agency and the Office of Naval Research.

[‡] Corresponding author.

The potential use of deception has been recognized in several areas, such as price negotiation [8, 9], multiobject auctioning [10], pursuit-evasion [11, 12], human relations [13], and card games [14]. In [8, 9], the authors analyze a negotiation where the players don not know each other's payoffs, but receive estimates from their opponents. In order to increase its gain each player may bias the estimate given. In [9], an advising scheme is proposed to make deception mostly useless. In [10], it is analyzed how a bidder can use deception to lower the price of an item sold in a multiobject auction. A pursuit-evasion game is analyzed in [11], where the evader corrupts the information available to its opponent to gain an advantage. It is assumed that the evader can jam the pursuer's sensor and therefore induce measurement errors, produce false targets, or interrupt the observations. Deception has also been studied in the context of military operations [11, 15, 16, 17, 18, 19]. A notable historical event was operation Overlord during the Second World War that culminated with the D-day invasion of France in June 1944 by the allied forces. The success of this operation relied heavily on deceiving the German command regarding the time and place of the sea-borne assault [19]. It is also widely recognized that, by the end of the cold war, the trend in Soviet naval electronic warfare was changing toward an independent type of combat action instead of a purely support role. This new role emphasized radio deception and misinformation at all levels of command [15]. The detection of false targets or decoys is now an important area of research in radar systems [16, 18].

In this paper, we analyze the use of deception in the framework of non-cooperative stochastic games with partial information. We take as a working example a prototype problem that arises in the control of military operations. In its simplest form, this game is played by an attacker that has to select one of several alternative targets and a defender that must distribute its defensive assets among them. This is a partial information game because the attacker has to make a decision without knowing precisely how the defense units have been distributed among the potential targets. We explore several variations of this game that differ in the amount of information available to the attacker. This can range from no information at all to perfect information provided, e.g., by intelligence, surveillance, or reconnaissance. The interesting cases happen between these two extremes because, in practice, the information available is not perfectly accurate and is often susceptible to manipulation by the opponent. It turns out that when the defender can manipulate the information available to its opponents—e.g., by camouflaging some of its defensive units and not camouflaging others—deception can be used to increase its payoff by effectively rendering the information available to the attacker useless.

The remaining of this paper is organized as follows. In Section 2, we formally introduce the simplest version of the prototype game where both attacker and defender have no information available to use in their decisions. This will serve as the baseline to compare the deception games that follow. In Section 3, we consider the extreme situation where the defender completely controls the information available to the attacker. We show that none of the players profits from this new information structure and deception is useless. This situation changes in Section 4 where the defender may profit from using deception. Paradoxically, when the degree of possible manipulation is high, deception becomes useless against an intelligent opponent since it will simply ignore the information that has potentially been manipulated. Section 5 contains some concluding remarks and directions for future research. A full version of this paper is available as a technical report [20].

2 A Prototype Non-Cooperative Game

Consider a game between two players that pursue opposite goals. The *attacker* must choose one of two possible targets (A or B) and the *defender* must decide how to better defend them. We assume here that the defender has a finite number of assets available that can be used to protect the targets. To make these assets effective, they must be assigned to a particular target and the defender must choose how to distribute them among the targets. To raise the stakes, we assume that the defender only has three defense units and is faced with the decision of how to distribute them among the two targets. We start by assuming that both players make their decisions independently and execute them without knowing the choice of the other player. Although it is convenient to regard the players as "attacker" and "defender," this type of games also arise in non-military applications. For example, the "attacker" could be trying to penetrate a market that the "defender" currently dominates.

The game described above can be played as a zero-sum game defined by the cost below, which the attacker tries to minimize and the defender tries to maximize:

$$J := \begin{cases} c_0 & \text{no units defending the target attacked} \\ c_1 & \text{one unit defending the target attacked} \\ c_2 & \text{two units defending the target attacked} \\ c_3 & \text{three units defending the target attacked.} \end{cases}$$

Without loss of generality we can normalize these constants to have $c_0 = 0$ and $c_3 = 1$. The values for the constants c_1 and c_2 are domain specific. Here, we consider arbitrary values for c_1 and c_2, subject to the reasonable constraint that $0 < c_1 \leq c_2 < 1$. Implicit in the above cost is the assumption that both targets have the same strategic value. We only make this assumption for simplicity of presentation.

As formulated above, the attacker has two possible choices (attack A or attack B) and the defender has a total of four possible ways of distributing its units among the two targets. Each choice available to a player is called a ***pure policy*** for that player. We will denote the pure policies for the attacker by α_i, $i \in \{1,2\}$, and the pure policies for the defender by δ_j, $j \in \{1,2,3,4\}$. These policies are enumerated in Tables 1(a) and 1(b), respectively. In Table 1(b), each "o" represents one defensive unit. The defender polices δ_1 and δ_2 will be called *3-0 configurations*, whereas the policies δ_3 and δ_4 will be called *2-1 configurations*.

policy	target assigned
α_1	A
α_2	B

(a) Attacker policies

policy	target A	target B
δ_1	o o o	
δ_2		o o o
δ_3	oo	o
δ_4	o	oo

(b) Defender policies

Table 1: Pure policies

The game under consideration can be represented in its *extensive form* by associating each policy of the attacker and the defender with a row and column, respectively, of a matrix $G \in \mathbb{R}^{2\times 4}$. The entry g_{ij}, $i \in \{1,2\}$, $j \in \{1,2,3,4\}$ of G corresponds to the cost J when the attacker chooses policy α_i and the defender chooses policy δ_j. The matrix G for this game is given by

$$G := \begin{array}{cc} & \begin{array}{cccc} \delta_1 & \delta_2 & \delta_3 & \delta_4 \end{array} \\ & \left[\begin{array}{cccc} 1 & 0 & c_2 & c_1 \\ 0 & 1 & c_1 & c_2 \end{array}\right] \begin{array}{c} \alpha_1 \\ \alpha_2 \end{array} \end{array} \tag{1}$$

In the context of non-cooperative zero-sum games, such as the one above, optimality is usually defined in terms of a saddle-point or Nash equilibrium [21]. A ***Nash equilibrium*** in pure policies would be a pair of policies $\{\alpha_{i^*}, \delta_{j^*}\}$, one for each player, for which

$$g_{i^*j} \leq g_{i^*j^*} \leq g_{ij^*}, \qquad \forall i, j.$$

Nash policies are chosen by rational players since they guarantee a cost no worst than $g_{i^*j^*}$ for each player, *no matter what the other player decides to do.* As a consequence, playing at a Nash equilibrium is "safe" even if the opponent discovers our policy of choice. They are also reasonable choices since a player will never do better by unilaterally deviating from the equilibrium. Not surprisingly, there are no Nash equilibria in pure policies for the game described by (1). In fact, all the pure policies violate the "safety" condition mentioned above for Nash equilibria. Suppose, for example, that the attacker plays policy α_1. This choice is certainly not safe in the sense that, if the defender guesses it, he can then choose the policy δ_1 and subject the attacker to the highest possible cost. Similarly, α_2 is not safe and therefore cannot also be in a Nash equilibrium pair.

To obtain a Nash equilibrium, one needs to enlarge the policy space by allowing each player to randomize among its available pure policies. In particular, suppose that the attacker chooses policy α_i, $i \in \{1,2\}$, with probability a_i and the defender chooses policy δ_j, $j \in \{1,2,3,4\}$, with probability d_j. When the game is played repeatedly, the expected value of the cost is then given by

$$\mathrm{E}[J] = \sum_{i,j} a_i g_{ij} d_j = a'Gd.$$

Each vector $a := \{a_i\}$ in the 2-dimensional simplex[1] is called a ***mixed policy for the attacker***, whereas each vector $d := \{d_j\}$ in the 4-dimensional simplex is called a ***mixed policy for the defender.*** It is well know that at least one Nash equilibrium in mixed policies always exists for finite matrix games (cf. Minimax

[1] We call the set of all vectors $x := \{x_i\} \in \mathbb{R}^n$ for which $x_i \geq 0$ and $\sum_i x_i = 1$, the *n-dimensional simplex.*

Theorem [22, p. 27]). In particular, there always exists a pair of mixed policies $\{a^*, d^*\}$ for which

$$a^{*\prime}Gd \leq a^{*\prime}Gd^* \leq a'Gd^*, \qquad \forall a, d.$$

Assuming that both players play at the Nash equilibrium the cost will then be equal to $a^{*\prime}Gd^*$, which is called the *value of the game.* It is straightforward to show that the unique Nash equilibrium for the matrix G in (1) is given by

$$a^* := \begin{bmatrix}\tfrac{1}{2} & \tfrac{1}{2}\end{bmatrix}', \tag{2}$$

$$d^* := \begin{cases} [\,\tfrac{1}{2}\ \tfrac{1}{2}\ 0\ 0\,]' & c_1 + c_2 \leq 1 \\ [\,0\ 0\ \tfrac{1}{2}\ \tfrac{1}{2}\,]' & c_1 + c_2 > 1 \end{cases} \tag{3}$$

with value equal to

$$a^{*\prime}Gd^* = \max\left\{\frac{c_1 + c_2}{2}, \frac{1}{2}\right\}.$$

This equilibrium corresponds to the intuitive solution that the attacker should randomize between attacking targets A or B with equal probability, and the defender should randomize between placing most of its units next to A or next to B also with equal probability. The optimal choice between 3-0 or 2-1 configurations (policies δ_1/δ_2 versus δ_3/δ_4) depends on the parameters c_1 and c_2. From (2) and (3) we conclude that 3-0 configurations are optimal when $c_1 + c_2 \leq 1$, otherwise the 2-1 configurations are preferable.

In the game described so far, there is no role for deception since the players are forced to make a decision without any information. We will change that next.

3 Full Manipulation of Information

Suppose now that the game described above is played in two steps. First the defender decides how to distribute its units. It may also disclose the position of some of its units to the attacker. On the second step the attacker decides which target to strike. To do this, it may use the information provided by the defender. For now, we assume that this is the only information available to the attacker and therefore the defender completely controls the information that the attacker uses to make its decision.

The rationale for the defender to voluntarily disclose the position of its units is to deceive the attacker. Suppose, for example that the attacker uses two units to defend target A and only one to defend B. By disclosing that it has units next to B, the defender may expect the opponent to attack A and, consequently, suffer a heavier cost.

In this new game, the number of admissible pure policies for each player is larger than before. The attacker now has 8 distinct pure policies available since, for each possible observation (no unit detected, unit detected defending target A, or unit detected defending target B), it has two possible choices (strike A or B). These policies are enumerated in Table 2(a). In policies α_1, α_2 the attacker ignores any available information and always attacks target A or target B. These policies are therefore called *blind.* In policies α_3 and α_4, the attacker never selects the target where it detects a defense unit. These policies are called *naive.* In policies α_5 and α_6 the attacker chooses the target where it detects a defending unit. These policies are called *counter-deception* since they presume that a unit is being shown close to the least defended target.

The defender has ten distinct pure policies available, each one corresponding to a particular configuration of its defenses and a particular choice of which units to disclose (if any). These are enumerated in Table 2(b), where "∘" represents a defense unit whose position has not been disclosed and "•" a defense unit whose position has been disclosed. Here, we are assuming that the defender will, at most, disclose the placement of one unit because more than that would never be advantageous. In policies δ_1 through δ_4 nothing is disclosed about the distribution of the units. These are called *no-information* policies. In policies δ_9 and δ_{10} the defender shows units placed next to the target that has fewer defenses. These are *deception* policies. Policies δ_5 through δ_8 are *disclosure* policies, in which the defender is showing a unit next to the target that is better defended.

This game can be represented in extensive form by the following 8×10 matrix

$$G := \begin{array}{c} \begin{array}{cccccccccc} \delta_1 & \delta_2 & \delta_3 & \delta_4 & \delta_5 & \delta_6 & \delta_7 & \delta_8 & \delta_9 & \delta_{10} \end{array} \\ \left[\begin{array}{cccccccccc} 1 & 0 & c_2 & c_1 & 1 & 0 & c_2 & c_1 & c_2 & c_1 \\ 0 & 1 & c_1 & c_2 & 0 & 1 & c_1 & c_2 & c_1 & c_2 \\ 0 & 1 & c_1 & c_2 & 0 & 0 & c_1 & c_1 & c_2 & c_2 \\ 1 & 0 & c_2 & c_1 & 0 & 0 & c_1 & c_1 & c_2 & c_2 \\ 1 & 0 & c_2 & c_1 & 1 & 1 & c_2 & c_2 & c_1 & c_1 \\ 0 & 1 & c_1 & c_2 & 1 & 1 & c_2 & c_2 & c_1 & c_1 \\ 1 & 0 & c_2 & c_1 & 0 & 1 & c_1 & c_2 & c_1 & c_2 \\ 0 & 1 & c_1 & c_2 & 1 & 0 & c_2 & c_1 & c_2 & c_1 \end{array}\right] \end{array} \begin{array}{c} \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \end{array} \tag{4}$$

policy	target assigned when ...		
	no obs.	unit detected at A	unit detected at B
α_1	A	A	A
α_2	B	B	B
α_3	B	B	A
α_4	A	B	A
α_5	A	A	B
α_6	B	A	B
α_7	A	B	B
α_8	B	A	A

(a) Attacker policies

policy	target A	target B
δ_1	○ ○ ○	
δ_2		○ ○ ○
δ_3	○○	○
δ_4	○	○○
δ_5	○ ○ ●	
δ_6		○ ○ ●
δ_7	○●	○
δ_8	○	○●
δ_9	○○	●
δ_{10}	●	○○

(b) Defender policies

Table 2: Pure policies

Just as before, for this game to have Nash equilibria one needs to consider mixed policies. However, this particular game has multiple Nash equilibria, one of them being

$$a^* := \begin{bmatrix}\frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}',$$

$$d^* := \begin{cases} [\,\frac{1}{2}\ \frac{1}{2}\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]' & c_1 + c_2 \leq 1 \\ [\,0\ 0\ \frac{1}{2}\ \frac{1}{2}\ 0\ 0\ 0\ 0\ 0\ 0]' & c_1 + c_2 > 1 \end{cases}$$

with value equal to

$$a^{*\prime} G d^* = \max\left\{\frac{c_1 + c_2}{2}, \frac{1}{2}\right\}.$$

This shows that (i) the attacker can ignore the information available and simply randomize among the two blind policies; and (ii) the defender gains nothing from disclosing information and can therefore randomize among its no-information policies. It should be noted that there are Nash equilibria that utilize different policies. For example, when $c_1 + c_2 > 1$, an alternative Nash equilibrium is

$$\bar{a}^* := \begin{bmatrix}0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & 0\end{bmatrix}', \qquad (5)$$

$$\bar{d}^* := \begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4}\end{bmatrix}'. \qquad (6)$$

In this case, the defender randomizes between deception and disclosure policies with equal probability and the attacker between the naive and counter-deception policies. However, in zero-sum games all equilibria yield the same value, so the players have no incentive to choose this equilibrium that is more complex in terms of the decision rules. Finally, it should also be noted that, because of the equilibrium interchangeability property for zero-sum games, the pairs $\{a^*, \bar{d}^*\}$ and $\{\bar{a}^*, d^*\}$ are also Nash equilibria [22, p. 28].

We have just seen that the attacker gains nothing from using the measurements available, even though these measurements give precise information about the position of some of the defense units. At an intuitive level, this is because the information available to the attacker is completely controlled by its opponent. And, if the defender chooses to disclose the position of some of its units, this is done solely to get an advantage. This can be seen, for example, in the equilibrium given by (5)–(6). We shall consider next a version of the game where the defender no longer has complete control over the information available to the attacker. For the new game, the attacker may sometimes improve its cost by using the available information.

4 Partial Manipulation of Information

In practice, when the defender decides to "show" one of its units it simply does not camouflage it, making it easy to find by the surveillance sensors used by the attacker. In the previous game we assumed that shown units are always detected by the attacker and hidden ones are not. We will deviate now from this ideal situation and assume that (i) shown units may not be detected and, more importantly, (ii) hidden units may sometimes be detected by the attacker. We consider here a generic probabilistic model for the attacker's surveillance, which is characterized by the conditional probability of detecting units next to a particular target, given a specific total number of units next to that target and how many of them are being shown. In particular, denoting by $\mathfrak{D}_A$ the event that defenses are detected next to target A, we have that

$$\mathrm{P}(\mathfrak{D}_A \mid \mathbf{n}_A = n_A,\ \mathbf{s}_A = s_A) = \chi(n_A, s_A),$$

where $\chi(\cdot)$ is the ***characteristic function of the sensor***, $\mathbf{n}_A$ denotes the total number of units defending target A, and $\mathbf{s}_A$ the number of these that are shown. We assume here that $\mathfrak{D}_A$ is conditionally independent of any other event, given specific values for $\mathbf{s}_A$ and $\mathbf{n}_A$. Since there is no incentive for the defender to show more than one unit $\mathbf{s}_A \in \{0,1\}$, whereas $\mathbf{n}_A \in \{0,1,2,3\}$. For simplicity of notation, we assume that the surveillance of target B is identical and independent, with

$$\mathrm{P}(\mathfrak{D}_B \mid \mathbf{n}_B = n_B,\ \mathbf{s}_B = s_B) = \chi(n_B, s_B),$$

where the symbols with the B subscript have the obvious meaning.

For most of the discussion that follows, the characteristic function $\chi(\cdot,\cdot)$ can be arbitrary, provided that it is monotone non-decreasing with respect to each of its arguments (when the other is held fixed). The monotonicity is quite reasonable since more units (shown or not) should always result in a higher probability of detection. However, it will be sometimes convenient to work with a specific characteristic function $\chi(\cdot,\cdot)$. One possible choice is:

$$\chi(n,s) = 1 - (1-p)^s(1-q)^{n-s}, \qquad (7)$$

$n \in \{0,1,2,3\}$, $s \in \{0,1\}$, where $0 \leq q \leq p \leq 1$. This model assumes that (i) the surveillance sensors will provide a positive reading (i.e., announce that units were detected next to a particular target) when they are able to detect, at least, one unit, and (ii) the probability of detecting a particular defense unit that is hidden is equal to q, whereas the probability of detecting a unit that is shown is equal to p. A few particular cases should be considered:

- When $\chi(n,s) = 0$, $\forall\ n,s$ (or when $p = q = 0$ in (7)) we have the first game considered in this paper, since the defense units are never detected.
- When

  $$\chi(n,s) = \begin{cases} 1, & s > 0 \\ 0, & s = 0 \end{cases} \qquad \forall n,s$$

  (or when $p = 1$ and $q = 0$ in (7)) the defense units are detected only when they are shown. This corresponds to the second game considered here, where the defender has full control of the information available to the attacker.

Another interesting situation occurs when placing more units next to a particular target makes that target more likely to be detected by the surveillance sensors *regardless of how many units are shown.* In such a case the attacker's sensors are said to be *reliable.* This can be formally expressed by the condition

$$\chi(n_1, s_1) \geq \chi(n_2, s_2), \qquad (8)$$

$\forall\ n_1 > n_2 : n_1 + n_2 = 3$, $\forall\ s_1, s_2$. Because the characteristic functions of the sensors are monotone non-decreasing with respect to each of its arguments (when the other is held fixed), it is straightforward to show that this reliability condition is equivalent to

$$\chi(2,0) \geq \chi(1,1). \qquad (9)$$

For the characteristic function in (7), this corresponds to

$$2q - q^2 \geq p.$$

We will see below that the attacker can choose naive policies, when the sensors are reliable. A special case of reliable sensors arises when $\chi(n,s)$ is independent of s for all values of n, (or when $p = q$ in (7)). In this case we have sensors that cannot be manipulated by the defender since the detection is independent of the number of units "shown" be the defender.

In terms of the policies available, the game considered in this section is very similar to the one in Section 3. The only difference is that, in principle, the attacker may now detect defense units next to both targets. In practice, this means that Table 2(a) should have a forth column entitled "units detected at A and B," which would result in 16 distinct pure policies. It turns out that not detecting any unit or detecting units next to both targets is essentially the same. Because of this we shall consider for this game only the 8 policies in Table 2(a), with the understanding that when units are detected next to both targets, the attacker acts as if no units were detected. It is not hard to prove that this introduces no loss of generality. The defender's policies are the same as in Section 3, and are given in Table 2(b).

This game can also be represented in extensive form by an 8×10 matrix. The reader is referred to [20] on how to construct this matrix. As before the game has Nash equilibria in mixed policies, but now the equilibrium policies depend on the values of $\chi(n,s)$.

To make the computation of the Nash Equilibrium simpler, we reduced the size of the matrix of the game using the intuitive notion that the optimal policies for the attacker and the defender will be symmetric. Then we found a Nash Equilibrium for the reduced game, and construct a solution for the actual game. We finally proved that this solution is in fact a Nash Equilibrium for the original game [20]. The optimal policies that we computed for this game are given in the following theorem:

Theorem 1. *1. For $\chi(2,0) \geq \chi(1,1)$, one of the Nash Equilibrium solutions for the players is*

$$a^* := \begin{bmatrix} 0 & 0 & \tfrac{1}{2} & \tfrac{1}{2} & 0 & 0 & 0 & 0 \end{bmatrix}',$$

$$d^* := \begin{cases} [\,\tfrac{1}{2}\;\tfrac{1}{2}\;0\;0\;0\;0\;0\;0\;0\;0\,]' & 1-\chi(3,0) \geq c_1 + c_2 - e_1 \\ [\,0\;0\;0\;0\;0\;0\;0\;0\;\tfrac{1}{2}\;\tfrac{1}{2}\,]' & 1-\chi(3,0) < c_1 + c_2 - e_1, \end{cases}$$

where $e_1 = (c_2 - c_1)\big(\chi(2,0) - \chi(1,1)\big)$.

2. For $\chi(2,0) < \chi(1,1)$ and $c_1 + c_2 \geq 1$, one of the Nash Equilibrium solutions for the players is

$$a^* := \begin{bmatrix} \tfrac{1}{2} & \tfrac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}',$$

$$d^* := \begin{bmatrix} 0 & 0 & \tfrac{e_2}{2} & \tfrac{e_2}{2} & 0 & 0 & 0 & 0 & \tfrac{1-e_2}{2} & \tfrac{1-e_2}{2} \end{bmatrix}',$$

where $e_2 := \frac{\chi(1,1)-\chi(2,0)}{\chi(1,1)-\chi(1,0)}$.

3. For $\chi(2,0) < \chi(1,1)$ and $c_1 + c_2 < 1$, one of the Nash equilibrium solutions for the players is

$$a^* := \begin{bmatrix} \tfrac{1-e_3}{2} & \tfrac{1-e_3}{2} & \tfrac{e_3}{2} & \tfrac{e_3}{2} & 0 & 0 & 0 & 0 \end{bmatrix}',$$

$$d^* := \begin{bmatrix} \tfrac{e_4}{2} & \tfrac{e_4}{2} 0 & 0 & 0 & 0 & 0 & 0 & \tfrac{1-e_4}{2} & \tfrac{1-e_4}{2} \end{bmatrix}',$$

where $e_3 := \frac{1-c_1-c_2}{\chi(3,0)-e_1}$ and $e_4 := \frac{e_1}{e_1-\chi(3,0)}$.

The proof of Theorem 1 can be found in [20].

Having computed the Nash Equilibrium solutions for the matrix, we can conclude the following:

1. When the sensors are reliable (i.e., $\chi(2,0) \geq \chi(1,1)$), the attacker randomizes among its naive policies and the defender either randomizes among the deception policies or the 3-0 no-information configurations. The latter occurs when the attacker only incurs in significant cost when 3 units are in its path and therefore 2-1 configurations are not acceptable for the defender. The value of the game is

$$a^{*\prime} G d^* = \frac{\max\{1-\chi(3,0), c_1 + c_2 - e_1\}}{2} \leq \frac{c_1 + c_2}{2}.$$

   This value for the cost is smaller than the one obtained in the previous two games, making it more favorable to the attacker, which is now able to take advantage of the surveillance information.

2. When the sensors are not reliable (i.e., $\chi(2,0) < \chi(1,1)$) and $c_1 + c_2 \geq 1$, the attacker randomizes among its blind policies and the defender randomizes between deception and no-information in 2-1 configurations. The probability distribution used by the defender is a function of the several parameters. However, the value of the game is always

$$a^{*\prime} G d^* = \frac{c_1 + c_2}{2}.$$

This means that the surveillance sensors of the attacker are effectively rendered useless by the defender's policy. This happens because the sensors are not reliable and therefore the defender can significantly manipulate the information available to the attacker. For the characteristic function of the sensors in (7), this occurs when the probability p of detecting a unit that is shown is significantly large when compared to the probability q of detecting a unit that is hidden. It is interesting to note that the region of the (p,q) parameter space where this happens is actually quite large (cf. Figure 1). This means that such situations are likely to occur in practice.

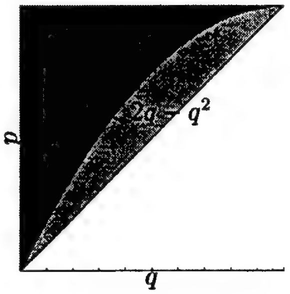


Figure 1: (p,q) Parameter Space

3. When the sensors are not reliable (i.e., $\chi(2,0) < \chi(1,1)$) and $c_1 + c_2 < 1$, the attacker randomizes between its blind and naive policies, whereas the defender randomizes between deception and no-information in 3-0 configurations. The value of the game is

$$a^{*\prime} G d^* = \frac{1}{2} - \frac{(1-c_1-c_2)\chi(3,0)}{2(\chi(3,0)-e_1)} \leq \frac{1}{2}.$$

Therefore, the attacker can attain a cost smaller than $\frac{1}{2}$ which would be obtained by only using blind policies.

Note that both in cases 2 and 3 the sensors are not reliable and the defender has sufficient power to manipulate the information available to the attacker so as to make it effectively useless. However, in case 3, the 2-1 configurations required for deception are very costly to the defender and deception is no longer a very attractive alternative.

5 Conclusions

We demonstrated that, when one of the players in a competitive game can manipulate the information available to its opponents, deception can be used to increase the player's payoff. We showed that an intelligent player can effectively render the information available to its opponent useless by carefully using deception. This study was carried out for a prototype problem in asset distribution in military operations. The ideas presented here can be applied to devise optimal strategies that use and counteract deception in many other problems. This is the subject of our current research.

References

[1] R. Bellman, *Dynamic Programming.* Princeton, NJ: Princeton University Press, 1957.

[2] S. Sorin and S. Zamir, ""Big Match" with lack of information on one side (III)," in T. E. S. Raghavan [23], pp. 101–112.

[3] C. Melolidakis, "Stochastic games with lack of information on one side and positive stop probabilities," in T. E. S. Raghavan [23], pp. 113–126.

[4] G. Kimeldorf, "Duels: An overview," in *Mathematics of Conflict* (M. Shubik, ed.), pp. 55–72, Amsterdam: North-Holland, 1983.

[5] P. Bernhard, A.-L. Colomb, and G. P. Papavassilopoulos, "Rabbit and hunter game: Two discrete stochastic formulations," *Comput. Math. Applic.*, vol. 13, no. 1–3, pp. 205–225, 1987.

[6] G. J. Olsder and G. P. Papavassilopoulos, "About when to use a searchlight," *Journal of Mathematical Analysis and Applications*, vol. 136, pp. 466–478, 1988.

[7] G. J. Olsder and G. P. Papavassilopoulos, "A Markov chain game with dynamic information," *Journal of Optimization Theory and Applications*, vol. 59, pp. 467–486, Dec. 1988.

[8] I. Greenberg, "The effect of deception on optimal decisions," *Operations Research Lett.*, vol. 1, pp. 144–147, Sept. 1982.

[9] S. Matsubara and M. Yokoo, "Negotiations with inaccurate payoff values," in *Proc. of the Int. Conf. on Multi-agent Syst.*, pp. 449–50, 1998.

[10] D. B. Hausch, "Multi-object auctions: Sequential vs. simultaneous sales," *Management Science*, vol. 32, pp. 1599–1610, Dec. 1986.

[11] Y. Yavin, "Pursuit-evasion differential games with deception or interrupted observation," *Computers and Mathematics with Applications*, vol. 13, no. 1–3, pp. 191–203, 1987.

[12] J. P. Hespanha, M. Prandini, and S. Sastry, "Probabilistic pursuit-evasion games: A one-step nash approach." Submitted to the 39th Conf. on Decision and Contr., Mar. 2000.

[13] C. Reimold, "Games of deception," in *Proc. of the Int. Professional Communication Conf.*, pp. 96–101, 1989.

[14] D. Billings, D. Papp, J. Schaeffer, and D. Szafrone, "Poker as a testbed for machine intelligence research," in *Advances in Artificial Intell.: Proc. of the 12th Biennial Conf. of the Canadian Soc. for Computational Studies*, pp. 228–238, 1998.

[15] M. Vego, "Soviet naval electronic warfare," *Signal*, vol. 44, no. 4, pp. 96–99, 1989.

[16] C. L. C. Oliveira, M. A. Grivet, and E. R. Pantoja, "Radar-ECM simulation system," in *Proc. of the Int. Microwave Conf.*, vol. 1, pp. 255–260, 1993.

[17] M. Tambe, "Recursive agent and agent-group tracking in real-time dynamic environment," in *Proc. of the Int. Conf. on Multi-agent Syst.*, pp. 368–375, 1995.

[18] S. Z. Yanglin, "Identification of false targets in bistatic radar system," in *Proc. of the IEEE National Aerospace and Electronics Conf.*, vol. 2, pp. 878–883, July 1997.

[19] R. W. Burns, "Deception, technology and the D-day invasion," *Engineering Science and Education J.*, vol. 4, pp. 81–88, Apr. 1995.

[20] J. P. Hespanha, Y. S. Ateşkan, and H. H. Kızılocak, "Deception in non-cooperative games with partial information," tech. rep., EE–Systems, University of Southern California, Los Angeles, CA, Apr. 2000.

[21] J. Nash, "Non-cooperative games," *Annals of Mathematics*, vol. 54, pp. 286–295, 1951.

[22] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory.* No. 23 in Classics in Applied Mathematics, Philadelphia: SIAM, 2nd ed., 1999.

[23] T. P. T. E. S. Raghavan, T. S. Ferguson, ed., *Stochastic Games and Related Topics: In Honor of Professor L. S. Shapley*, vol. 7 of *Theory and Decision Library, Series C, Game Theory, Mathematical Programming and Operations Research.* Dordrecht: Kluwer Academic Publishers, 1991.

Multi-Model Predictive Control: From Air Operations to Enterprise Optimization

Luis Rabelo, Datta Godbole, Jan Jelinek, Vipin Gopal
Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418
Contact: luis.rabelo@honeywell.com, phone (612) 951-7131

Abstract[1]

For the modern enterprise, growth has become an important concern. With the effects of recent re-organizations, the emphasis on "do more with less," increasing competition, the lifetimes of products shortened due to a higher pace of technological change, uncertainties in demand and the complex and conflictive economic forces due to globalization, the development of effective growth strategies is becoming a very difficult challenge. However, it is very clear that the ability of the modern enterprise to allocate resources is critical to its growth strategy. The development of planning models, organizational learning, and experimentation nurture this ability. This paper reviews the on-going efforts to use multi-model predictive control (M^2PC) technology to provide new ways to optimize the modern enterprise and promote organizational learning in order to achieve growth. M^2PC technology expands the traditional model predictive control scheme, with large-scale, more complex models and optimization abilities. This paper illustrates with two case studies, one in project selection and the other one in supply chain management, the possibilities of the applications of M^2PC in enterprise optimization.

1. Introduction

The analogy between military operations (e.g., air operations by the Joint Forces Air Component Commander (JFACC)) and corporate decisions is very appealing. For instance, in air operations, JFACC [1] control actions consist of allocating resources (wings, squadrons, air defense systems, AWACS) to different geographical locations in the theater, defining a sequence of tasks for the aerospace systems at each location, and providing feedback control for the execution of these tasks in the presence of uncertainties and a hostile enemy. In a similar fashion, managers control investment actions today in new marketing programs, or in R&D, or even in capital expenditures, to generate the possibility of new products or new markets, or develop new technological processes in the presence of uncertainties (e.g., market demand, global forces) and "hostile" competitors.

The novel multi-model predictive control (M^2PC) method is aimed at dramatically improving the agility and stability of military air operations. M^2PC was obtained by enhancing the models and optimization algorithms utilized in traditional Model Predictive Control (MPC) systems. This paper discusses the possible utilization of M^2PC to improve the resource allocations in the modern enterprise and thereby stimulate growth. Two case studies are used to bridge the gap between the "industrial" world and the concepts associated with model predictive control, hybrid models, game theory and probabilistic system analysis.

2. The Framework of M^2PC

M^2PC technology is being developed to help the JFACC planning and control system achieve agile and stable control of military operations (Figure 1). The M^2PC system is based on the core technologies of model predictive control, hybrid systems, game theory, and probabilistic analysis using randomized algorithms.

MPC is an optimal control method that uses a plant model to predict the effect of an input profile on the evolving state of a plant. At each step, an optimal control problem is solved and the optimal profile is implemented until another plant data sample becomes available. The updated plant information is used to solve a new optimal control problem and the process is repeated. MPC has succeeded far more than traditional and modern control approaches in handling delays and constraints.

[1] This document is based upon work supported by DARPA JFACC program through SPAWAR Systems under the contract no. N66001-99-C-8501. Any opinions, findings, conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of DARPA and SPAWAR Systems.

For the military application, we have developed predictive models of battle dynamics [2,3,4]. Since battle is inherently stochastic, the models provide an evolution of probability distribution over the state of the system as a function of time. These models are a function of number of units in a force, units' effectiveness and feedback control structure. The models are used to develop a model predictive controller to calculate the optimal deployment of resources on a battlefield. The M^2PC framework enhances this basic MPC system by adding multiple models changing the rules for switching between different battle strategies and the identification of enemy strategies. Given the stochastic nature of battle, randomized algorithms [5] are used to conduct a simulation-based approximation of the optimal deployment of forces.

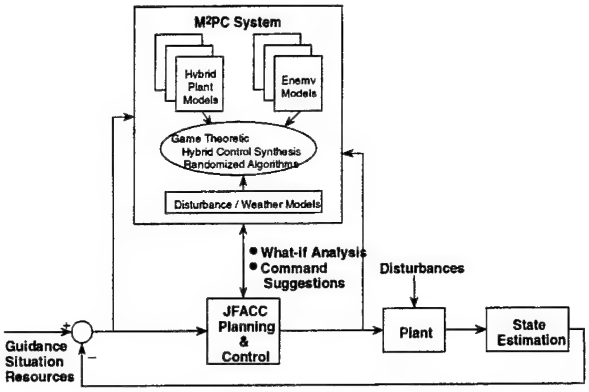


Figure 1. M^2PC system supports JFACC in achieving agile and stable control of military operations. The M^2PC system is capable of analysis as well as optimal control synthesis for the JFACC air campaign.

Similar problems arise in the management of large enterprises [6,7,8,9,10]. In a multinational company with multiple business units, the managers are faced with resource allocation problems akin to battle commanders. They have to also develop a strategy consisting of a sequence of modes such as investments in R&D, marketing and new production facilities. Finally the outcomes of their actions are stochastic because of the uncertainties of the marketplace, strategy execution and competitors' strategies.

M^2PC might be applied to business processes in a very similar fashion to how it is applied to military processes. Multiple predictive models of enterprise dynamics will be built using the cause-effect relationships of business dynamics. These models will be hybrid in nature, containing both continuous dynamics and logical/algebraic constraints [11,12,13,14,15]. Our game theoretic and randomized optimization algorithms will then be used to identify optimal strategies for growth. The M^2PC system can also be used to rank different options qualitatively and provide what-if analysis capability.

3. Case Study 1: Project Selection with Investment Opportunities

The classical managerial practice for project selection is to calculate the present value of (1) the expected cash flows that the investment will generate, and (2) the expenditures required to undertake the project. Then, the net present value (NPV) is determined. If NPV is greater than zero, the manager should go ahead and invest.

However, there are some problems with the NPV rule, especially when it is applied to investment opportunities. These are explained as follows [16]:

1. *NPV is based on faulty assumptions* - It assumes one of two things: the investment is reversible; or, if the investment is irreversible, it is a now-or-never proposition.
2. *NPV ignores the value of creating options* - Sometimes investments create options that enable the company to undertake other investments in the future should market conditions turn favorable.
3. *Uncertainty plays a minor role in the NPV rule* – Uncertainty is not central in the NPV rule. It is only somewhat "added" in the calculation of the discount rate used to compute present values.

Economists [16,17,18,19,20] have started to believe that thinking of capital investments as options changes the theory and practice of decision making. The option theory of investment helps to overcome several of the deficiencies of the traditional NPV rule. However, a question still remains unsolved:

How does one determine the expected stream of profits that the proposed project will generate and the expected stream of costs required to implement the project, taking into consideration the volatility and unpredictability of the real-world?

We think that some of the concepts generated from M^2PC can help in the application of the option theory of investments to project selection. One of these concepts is that of predictive models.

3. 1. Predictive Models

Our JFACC program has been able to develop methodologies to build predictive models of battles. These methodologies take into consideration the random nature of weapon effects, enemy behavior, and real time information assessment effectiveness. There is an interesting observation to take into account: the cost of reducing uncertainty escalates as we approach 0. As expressed by Jelinek [4], "The lesson to learn from this observation simply is that for stochastic systems (plants) like battles the task objective cannot be meaningfully stated without a desired certainty qualification, because there is no absolute certainty in combat."

Battle planning could be modeled with Monte Carlo simulation in order to provide the required information for planning. However, this approach is impractical due to the number of simulations required to obtain reliable results. A better approach is to develop models that predict the battle state probability distributions. Then, it is possible to use these predictive models for battle planning formulations.

The process of developing the models follows the modeling concept depicted in Figure 2. A battle, due to its ever-changing structure, needs models that must be continuously rebuilt on-line. The state variables **X** of the models are random. On the other hand, the internal model is deterministic with its state variables representing various statistics (**Xbar**). An auxiliary Monte Carlo simulation model can be used for the conversion from **X** to **Xbar**. The Monte Carlo simulation model can be used to develop estimates of the expectations, variances and other statistics, and then directly compared with those produced by the internal model to check its validity. This methodology imposes rigorous construction rules whose observance will guarantee that our simulator is indeed congruent with the plant (even though there is no way to confirm this directly by comparing the inputs and outputs from the simulator and the stochastic plant).

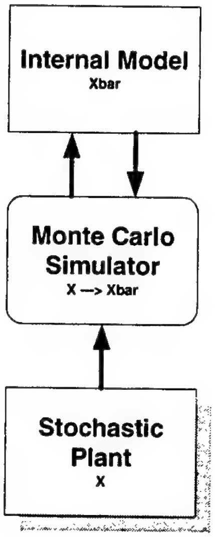


Figure 2. Modeling methodology to build predictive models in M^{2}PC.

As expressed by Jelinek[2], "Once we obtain the experimental data, **Xbar**, from the Monte Carlo simulation, we can follow the model building approach usual in industrial control and fit them with a solution of a suitable set of differential or difference equations, whose parameters are numerically optimized to guarantee the best approximation." In this approach, the concepts of systems dynamics can support the initial developments of the internal models and this will be explained below in the Case 2. This methodology has produced models that have proven superior to classical attrition models like the Lanchester equations [2,4].

Now that the predictive model (i.e., internal model) has been developed, it can accept as input parameters the number of attack airplanes, the number of decoys, the number of attack airplanes and decoys from the enemy, the lethality of the attack airplanes, and the lethality of the enemy attack airplanes. When exercised, the predictive model generates a sequence of probability distributions of possible battle states after repeated missions. In the sequence illustrated in Figure 3, the plot densities are proportional to the probabilities. The outcome of the first mission shows that the most likely number of survivors will be two or three blue attack airplanes and seven red attack airplanes, but other outcomes are possible. To produce this sequence, the specific rule of engagement and the effectiveness of the real time damage assessment information that the commander receive were selected. Using M^{2}PC optimization, it is possible to obtain the optimal deployment of attack airplanes needed to guarantee success and its probability [21], and minimize losses.

Now, the question to answer is how this concept will complement the utilization of the NPV rule. We will use a case to explain this.

3.2. Dealing with Real-World Uncertainties in Project Selection

The following case is based on the Specialty Additives Division (Specialty Chemicals Segment) of a Fortune 200 Global Enterprise (and also inspired by [19] - some of the information has been disguised to protect the proprietary interests of the company). The Specialty Additives Division sells ingredients, thickeners, and additives for pharmaceuticals, personal care, and home care markets. This division has several manufacturing plants geographically distributed: Kentucky and Korea. This division in 1993 was considering building a new plant immediately in Belgium to expand into the European Personal Care Market ("Phase 1"). The division's managers anticipated further investments in 1997 (almost double the initial investment), to expand the plant's capacity ("Phase 2"). The initial investment creates the opportunity for subsequent growth. Using traditional NPV, the project will be very difficult to justify. This expansion opportunity has considerable option value because the initial investment buys the right to expand (or not) in 1997.

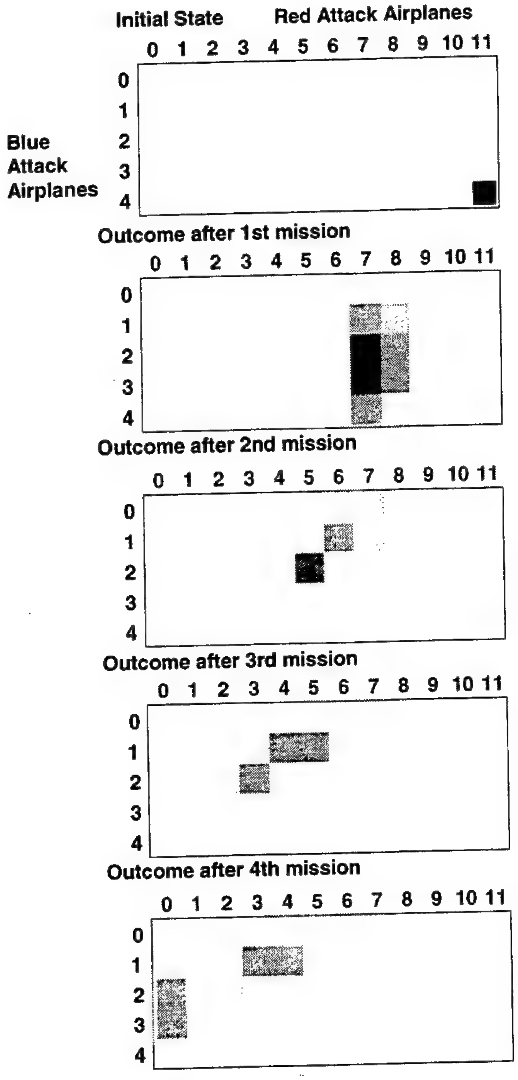


Figure 3. State probability distribution of a battle after 4 missions.

Luerhman [19] has explained a step by step framework to deal with this type of project. The approach emphasizes:

NPV (entire proposal) = NPV (Phase 1 assets) + Call Value (Phase 2 assets)

One of the first steps is to map the project's characteristics onto call option variables. The present value of the assets acquired when and if the division exercises the option is by analogy equal to the stock price of the option. The exercise price of the option is represented by the expenditures required to acquire the Phase 2 assets. In this case, the time to expire is four years. The four-year risk-free rate (r_f) of interest has to be obtained by studying U.S. Government Bonds and other similar type of securities. As with any other project, the risk-adjusted discount rate can be obtained by studying the Capital Asset Pricing Model (CAPM) [22]. In addition, the weighted average cost of capital (WACC) offers a good approximation ("as long as the company's projects do not differ greatly from one another in their nondiversifiable risk." [16]).

The next value to be obtained is the standard deviation of the returns (σ). The historical variance of stock market returns can be calculated from n observations by using

$$\sigma^2 = n/(n-1)\ \Sigma_t\ (r_t - \text{average return } r)^2/n$$

where r_t is the rate of return on day t and defined as the measure of the rate of return of the stock from t – 1 to time t. However, the calculation of σ for a project is not straightforward. Luerhman [19] recommends approaches such as educated guesses, the use of historical data, the study of the current prices of options traded on organized exchanges, and the use of Monte Carlo simulations.

The next steps of Luerhman's framework are to separate Phase 1 from Phase 2 and obtain the present value of the assets acquired when the division exercises the option and the expenditures required to acquire the Phase 2 assets. In addition, the value of Phase 1 (i.e., the NPV of Phase 1) becomes *at least* the value of the project.

The final steps involve solving the Black-Scholes model and obtaining the NPV for the entire proposal. The Black-Scholes model is represented by [22,23,24]

$$C = S\,N(d_1) - L\,e^{-r_f t}\,N(d_1 - \sigma t^{0.5})$$

where

$$d_1 = (\ln(S/L) + (r_f + \sigma^2/2)t)/\ \sigma t^{0.5}$$

and where

C = The call option value of the project (Phase 2).
S = The present value of the assets acquired when the division exercises the option (Phase 2).
N(d) = The probability of a random draw from a standard normal distribution will be less than d. N(d) can be viewed as "risk adjusted probabilities that the call options will expire in the money." [23]
L = expenditures required to acquire the Phase 2 assets.
t = time to maturity of option (Phase 2), in years.

Now that the value of the expansion (Phase 2) has been obtained, the NPV for the entire proposal can be determined.

3.3 Using M^2PC to help the handling of uncertainties

M^2PC can contribute to the project selection process. Predictive models can be developed by using the approach outlined in the section on Predictive Models to estimate the expected stream of operating profits from the investments in the Specialty Additives Division. These models will include the behaviors of the market, the competitors, and other environmental and process risks. The concepts included in M^2PC such as randomized algorithms, probabilistic hybrid systems, game theory, and multi-agent systems will be very important for the development of these models.

The managers of the Specialty Additives Division use a consensus projection. These estimates can be used as references to start building the models. However, the predictive models do not need to use the "faulty assumption" that Phase 2 will begin at a fixed point in time. Furthermore, the predictive models can take into consideration factors such as changes in the risk-free interest rate and the volatility of the present value of Phase 2. These factors are assumed constant by the Black-Scholes pricing formula.

The predictive models can generate a sequence of probability distributions of possible profit states of the Specialty Additives Division after repeated business cycles (i.e., fiscal years) with different investment levels and different competitors' reactions and market responses. This will allow for a better estimation of cumulative volatility ($\sigma t^{0.5}$). The predictive models can provide a better measure of the changes in variance over time. The modeling of costs (losses) can be incorporated in our models (e.g., "companies trying to be first to market with the next generation of a hot product will incur large costs if deferral allows a competitor to preempt them" [19]).

Again, the predictive models do not tell us how to execute a task in the optimal way. Applying hybrid optimization techniques, M^2PC can answer questions such as:

1. Whether to invest?
2. When to invest and the "size" of the investments (i.e., the optimal schedule of investments)? This will be one of the most important answers!

4. Case Study 2: Supply Chain Management

A *supply chain* is a network [25] of facilities and distribution options that performs several operations such as the procurement of materials, the transformation of these materials into intermediate and finished products, and the distribution of these finished products to customers. *Supply Chain Management* is the synchronization of the supply chain in order to optimize the creation of value for shareholders and customers. The *goals* of supply chain management are [25,26,27,28]

1. To reduce inventory
2. To increase customer service
3. To increase profits

The following case is based on the Specialty Additives Division (Specialty Chemicals Segment) of a Fortune 200 Global Enterprise introduced in the Case 1: Project Selection with Investment Opportunities. The supply chain of the Specialty Additives Division is a good example of how ideas from MPC can be used to improve strategy and operations by generating new optimal policies.

4.1 System Dynamics and Model-Based Optimization

Over the last decade, the specialty chemicals industry has experienced slower growth and lower overall profitability within a more competitive environment than in the preceding decade. Therefore, an increased awareness of process improvements and a reassembly of supply chains around new and changing business models was expected.

In 1999 some senior managers were concerned about the impact of the inventory's oscillations on profits. They approved the installation of an expensive Enterprise Resource Planning (ERP) system in 1996. The ERP system provides instantaneous information about the levels of inventories globally. The ERP system has reduced the information uncertainty and integrated the different accounting and financial models. However, the Director of Supply Chain at headquarters (Richmond, VA) has not been able to control the oscillations in inventories even though demand has been almost constant in the last 6 months (Figure 4).

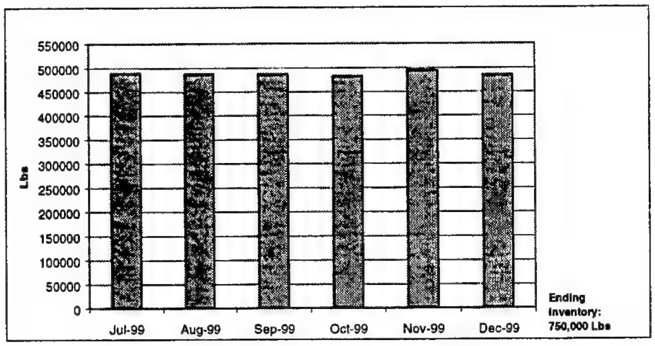


Figure 4. Demand for the Specialty Additives Division (July 1999 – December 1999).

The cost structure of the Specialty Additives Division includes relatively low manufacturing costs and high gross profit margins but also high marketing and technical service costs. Specialty chemical companies are trying to differentiate themselves not only with product innovations but also with greater levels of customer service, including delivering the right product on time. Inventory coverage of about 3 weeks *("evolved safe solution")* is thought to provide a good balance between the selection available to customers and carrying costs. Low coverage and backlogs hurt sales and customer relationships (a cost difficult to quantify but with disastrous business consequences in the specialty chemicals market). On the other hand, high inventories slash profits as carrying costs increase. The Specialty Additives Division has an adjustment time of approximately 3 months to re-organize a factory and fine-tune the supplier and transportation networks. The adjustment time for inventory decisions is approximately 2 weeks. In addition, the growth rate is forecast to decline to an average of 2.5% with a very similar demand pattern in 2000.

System dynamics [29,30,31,32,33] concepts can be used to start developing a basic model of the desired supply chain. System dynamics includes a variety of tools and concepts to support the knowledge elicitation process, to help to communicate the boundary of a model, and to represent causal structures (i.e., underlying cause and effect relationships and connections between the components of a system). These tools include model boundary diagrams, subsystem diagrams, causal loop diagrams and stock and flow maps [33].

Using system dynamics, we are able to build a model [34,35] (simplified for illustration purposes; a more realistic model would require more details):

Supply Chain Configuration = $\int$***Adjusting Supply Chain Configuration [Supply Chain Components]***

Inventory = $\int$***(Production - Sales) [Lbs]***

Expected Demand = $\int$***(Factor2-Factor1) [Lbs/Month]***

Initial Demand = 500000 [Lbs/Month]

Demand = Initial Demand + Demand Change [Lbs/Month]

Factor1 = Expected Demand / Expectation Formation Time [Lbs/(Month^2***]***

Factor2 = (Initial Demand + Demand Change) / Expectation Formation Time [Lbs/(Month^2***]***

Adjustment Time for Inventory = 0.5 [Months]

Production Desired = Expected Demand + Δ***Inventory [Lbs/Month]***

Supply Chain Configuration Desired = Production Desired / Production Effectiveness Factor [Supply Chain Components]

Adjustment Time for Supply Chain Configuration = 3 [Months]

Adjusting Supply Chain Configuration = (Supply Chain Configuration Desired – Supply Chain Configuration)/Adjustment Time for Supply Chain Configuration [Supply Chain Components/Month]

Δ***Inventory = (Inventory Desired -Inventory)/Adjustment Time for Inventory [Lbs/Month]***

Production = Supply Chain Configuration * Production Effectiveness Factor [Lbs/Month]

Sales = Demand [Lbs/Month]

Expectation Formation Time = 0.5 [Months]

Figure 5 shows the "predicted" ending inventory levels for the period of Jan-00 to June-00 for the Specialty Additives Division. Analysis of the system using simple control theory concepts shows that, to avoid oscillations, the relationship of the ***Time to Adjust Inventory*** and the ***Time to Adjust Supply Chain Configuration*** should be expressed by

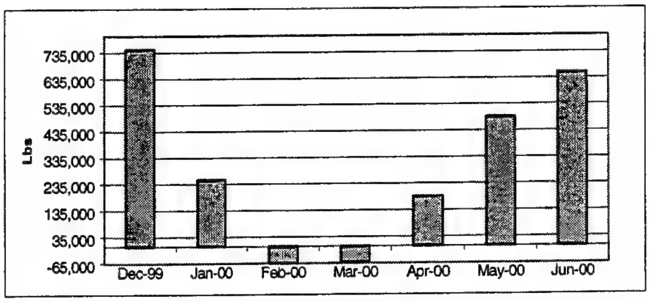


Figure 5. Predicted ending inventory levels (January 2000 – June 2000) by the system dynamics model.

4 Adjustment Time for Supply Chain Configuration > Adjustment Time for Inventory

Now, the model can be optimized following the objectives outlined by the financial and operational plans of the firm (Figure 6). This optimization provides the guidelines for the solutions to be implemented by the Specialty Additives Division. Details about the supply components can be added to the model and the *"evolved safe solution"* level of inventory can be challenged with a more optimized one.

It is very clear that the driver behind supply chain management is to remove inefficiencies, excess costs and excess inventories from the supply pipeline. Supply chain management requires multiple models to represent very well this supply pipeline with its risks, uncertainties, delays in decision making and execution, and constraints. In addition, the dynamics of a supply chain contains both continuous and discrete variables. The optimization of these models must meet the customers' demands and specific requirements under uncertainties.

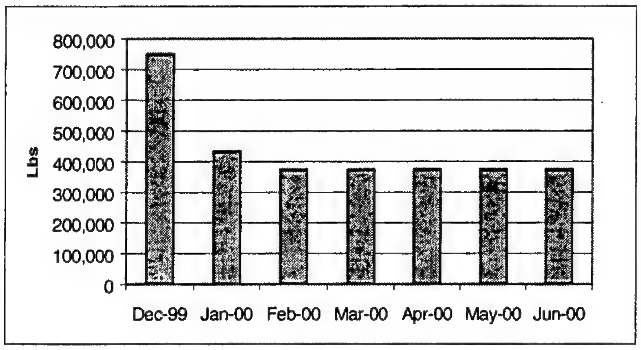


Figure 6. Effects of the optimal supply chain configuration on ending inventories with respect to the "evolved" safe inventory level (approximately 375,000 Lbs).

Using concepts developed from our JFACC program, we can develop models which are probabilistic in nature to allow calculations of the expected answers of the systems, their variances and other statistics over arbitrary long prediction horizons. These models will be complementary to the traditional system dynamics approach (i.e., system dynamics models are a good starting point). This will depend on the uncertainty level assessed in particular situations. Again, we can solve these predictive models to develop policies and establish critical parameters for the supply chain (e.g., inventory levels, the number of factories to be acquired). However, M^{2}PC adds a new dimension: operational supply chain management.

4. 2. Operational Supply Chain Management

The previous section discussed the optimal configuration of a supply chain for avoiding oscillations in inventories before execution starts. It is well known that demand will change, and that uncertainties in the network of outsourcers, and other scenarios, will arise during the business cycle. It is possible to use our predictive models to build a model predictive controller of the supply chain operations (Figure 7). As expressed by Jelinek [4], "Prior to the first mission, both the no-control and MPC-control strategies do the same calculations, namely compute the optimal number of aircraft to be flown in the first round (mission) using the initial deployment optimizer." This is analogous to our problem in the supply chain: initially, using an open loop scheme, the optimal supply chain configuration is provided taking into consideration a defined horizon. However, after the first period, the schemes will start to differ. The M^{2}PC controller will first look at the supply chain assessment and critically reevaluate its situation from both local and global performance (due to the multiple models strategy). "For this purpose he/she will use the same model as before, but it will enter the intelligence updates on the actual enemy strength after the first mission and will also reduce by one the maximum number of missions allowed to fulfill the task objective." [4] This reassessment will produce some corrections to the resource allocation decisions, inventory levels, and production schedules. "Feedback is thus closed through the ongoing replanning and implemented as corrections to package composition." Again, the notions of reachability, safety, and stability are very important for operational supply chain management.

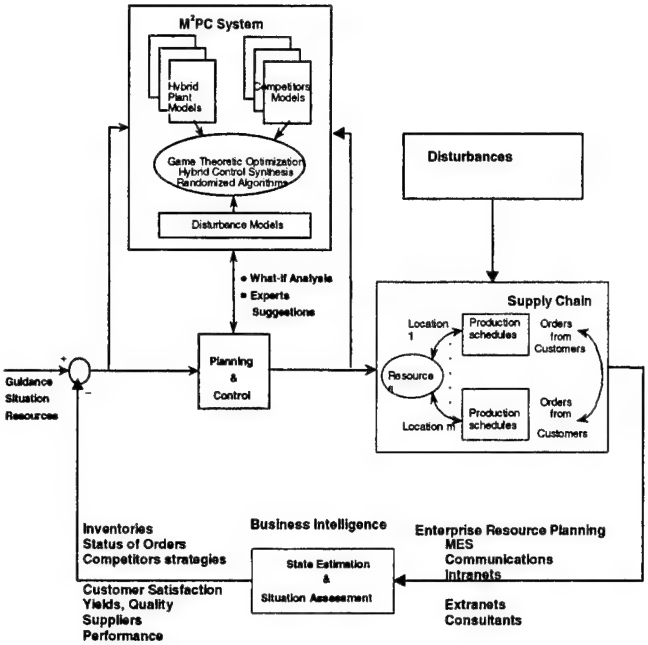


Figure 7. Operational supply chain management concept using M^{2}PC. M^{2}PC can control the supply chain ("a large-scale multi-agent distributed system") in the presence of uncertainties, market disturbances, and competitors.

5. Final Discussion

Ideas from M^{2}PC can have implications on the support of the decision making process (e.g., the project selection process) and on the design of agile supply chains (e.g., reducing the number of steps, de-constructing the value chain, and de-bottlenecking) in the modern enterprise.

"Every battle is a particular realization of the random processes involved, and if the combatants had a chance to fight it over and over under the same rules of engagement, the outcome would vary from one run to another." [2] The same can be said of the business world: business strategy is more like a series of options. Our JFACC program has been able to develop models of battles. This methodology can be extended to a simulation of the business environment to capture the uncertainties and the different risks. M^2PC can be used to guide future investment decisions.

Supply chain management requires multiple models to represent the modern enterprise (e.g., financial, transportation, information for decision-making risks, and operational models). The optimization of these models must meet the customers' demands and specific requirements under uncertainties. In addition, supply chain management requires organizational flexibility and responsiveness, and internal and external adjustments. Strategic, tactical, and operational aspects are very important. M^2PC can be used to generate policies and re-design the supply chain effectively to synchronize the network of raw materials sourcing with the making the product and the delivery through the distribution networks and to the customer.

In future reports, we will be discussing in more detail some of our current initiatives to apply M^2PC to enterprise optimization problems.

Bibliography

1. Joint Publication 3-56.1, "Command and Control for Air Operations."
2. Jelinek, J. "Predictive Models of Battle Dynamics," JFACC project report, Honeywell Technology Center, February 2000.
3. Jelinek, J. "Predictive Models of Battle Dynamics: Part II – Delayed Feedback," JFACC project report, Honeywell Technology Center, April 2000.
4. Jelinek, J. "Model Predictive Control of Battle Dynamics," JFACC project report, Honeywell Technology Center, May 2000.
5. Vidyasagar, M., *A Theory of Learning and Generalization with Applications to Neural Networks and Control Systems*. Berlin: Springer-Verlag, 1997.
6. Holmstrom, B. "Agency Costs and Innovation," *Journal of Economic Behavior and Organization*, 12, pp. 305-327, 1989.
7. Lygeros J., D. Godbole, and S. Sastry, "Multi-agent Hybrid System Design using Game Theory and Optimal Control," in Proceedings of IEEE Conference on Decision and Control, Kobe, Japan, pp.1190-1195, 1996.
8. Pujet, N. and E. Feron, "Modeling an Airline Operations Control," in Proceedings of 2nd USA/Europe Air Traffic Management R&D Seminar, December 1998.
9. Dhaenens-Flipo, C., A. Landrieu, Z. Binder, and G. Finke, "Coordination of Manufacturing and Distribution: Two Industrial Examples," in Proceedings of 14th World Congress of the IFAC, pp. 343-347, 1999.
10. Kosanke, K. and J. Nell, "Enterprise Organisations and the new enterprise paradigms", in Proceedings of 14th World Congress of the IFAC, pp. 139-144, 1999.
11. Rawlings, J. and K. Muske, "The Stability of Constrained Receding Horizon Control," *IEEE Trans. Aut. Control,* 38(10): 1512-1516 (1993)
12. Basar T., and G. Olsder, *Dynamic Non-cooperative Game Theory*, Academic Press, 1995.
13. De Alfaro, L. "Formal Verification of Probabilistic Systems," Ph. D. Dissertation, Department of Computer Sciences, Stanford University, 1997.
14. Gopal, V. and L. Biegler, "Large Scale Inequality Constrained Optimization and Control," IEEE Control Systems Magazine, 18(6), 57-68 (1998)
15. Bemporad, A. and M. Morari, "Control of systems integrating logic, dynamics and constraints," *Automatica,* 35(3), 1999.
16. Dixit, A. and R. Pindyck, "The Options Approach to Capital Investment," *Harvard Business Review*, pp. 105-155, May-June 1995.
17. Kester, W., "Today's Options for Tomorrow's Growth," *Harvard Business Review*, pp. 153-160, March-April 1984.
18. Luerhrman, T., "What's It Worth: A General Manager's Guide to Valuation," *Harvard Business Review*, pp. 132-142, May-June 1997.
19. Luerhrman, T. "Investment Opportunities as Real Options," *Harvard Business Review*, pp. 3-15, July-August 1998.
20. Luerhrman, T., "Strategy as a Portfolio of Real Options," *Harvard Business Review*, pp. 89-99, September-October 1998
21. Gass, N., "Analytical model for close combat dynamics," *Journal of Operations Research Society*, Vol 48, No 2, pp. 132-141, 1997.
22. Brealey, R. and S. Myers, *Principles of Finance*, Upper Saddle River, N.J.: McGraw-Hill, 2000.
23. Bodie, Z., A. Kane, and A. Marcus, *Investments*, Boston: McGraw-Hill, 1999.
24. Bodie, Z., P. Samuelson, and R. Merton, *Finance*, Prentice Hall, 1999.
25. Montgomery, S. "Semiconductor Supply Chain Management," FAB International, pp. 79-82, 1999.
26. Bovet, D. and J. Martha, *Value Nets*, John Wiley & Sons, 2000.
27. AMR Research, Supply Chain Strategies Outlook (2000), Report, February 2000.
28. Forrester Research Group (2000), B2B Insights: eMarketPlaces, Report, February 2000.
29. Forrester, J. *Industrial Dynamics*, Cambridge: MIT Press, 1961.

30. Forrester, J., "Nonlinearity in High-Order Models of Social Systems," Workshop on Modelling Complex Systems, 1985.
31. Senge, P. *The Fifth Discipline*, New York: Currency Doubleday, 1990.
32. Morecroft, J. and J. Sterman, *Modeling for Learning Organizations*, Portland, Oregon: Productivity Press, 1994.
33. Sterman, J. *Business Dynamics*, Boston: McGraw-Hill, 2000.
34. Hines, J., "Notes on Modeling," Systems Dynamics II, MIT Sloan School of Management, Fall 1999.
35. Hines, J. and P. Goncalves, "Workforce Inventory Oscillator," Systems Dynamics II, MIT Sloan School of Management, Fall 1999.

Sequential Linear Quadratic Method for Differential Games

H. Mukai,[*] A. Tanikawa, I. Tunay, I. N. Katz, H. Schättler,
P. Rinaldi, I. A. Ozcan, G. J. Wang, L. Yang and Y. Sawada[†]
Department of Systems Science and Mathematics
Washington University

Abstract

In this paper, we present a numerical method for computing a Nash solution to a zero-sum differential game for a general nonlinear system based on a sequential linear-quadratic approximations. The technique is used to design a game-theoretic controller. Numerical results are given which show the performance of the method as well as the performance of the resulting controller under noisy observations and model mismatch in parameters.

1. Introduction

Let U denote the set of $\mathbb{R}^m$ - valued continuous functions on $[t_0, t_f]$. Consider a system governed by the ordinary differential equation,

$$\frac{d}{dt}x(t) = f(x(t), u(t)),\ t \in [t_0, t_f];\ \ x(t_0) = z_0, \qquad (1)$$

where $f(x, u)$ is an $\mathbb{R}^n$-valued C^1-class function on $\mathbb{R}^n \times \mathbb{R}^m$. Given any control $u \in U$ and an initial state $x(t_0) = z_0$, we assume that equation (1) defines a unique continuously differentiable solution $x(t), t \in [t_0, t_f]$, which is called the *trajectory* of the system produced by control u and denoted also by $x[u] \in X$, where X is the space of continuously differentiable $\mathbb{R}^n$-valued functions on $[t_0, t_f]$.

We consider the following game situation. The control function u consists of two parts, u^B and u^R, corresponding to the two forces, the *Blue* and the *Red*: $u = (u^B, u^R)$. As a cost function, we consider

$$J(u) = \int_{t_0}^{t_f} g(x(t), u(t))dt + g_f(t_f, x(t_f)). \qquad (2)$$

However, it is often more convenient to consider $J(u)$ as a function of both u and x with an additional constraint (1) connecting u and $x = x[u]$, i.e., $J(u) = \tilde{J}(x[u]; u)$. Our objective is to solve

$$\min_{u^B} \max_{u^R} \Big\{ \tilde{J}(x; u^B, u^R) \ \Big|\ \frac{d}{dt}x(t) = f(x(t), u(t)), \\ x(t_0) = z_0 \Big\}. \qquad (3)$$

In Section 2, for computing a solution to the game problem (3), we will propose an iterative method, whose i-th subproblem is obtained from the original problem by linearizing the differential equation (1) around the i-th approximate solution (u_i, x_i) and expanding the cost function J to the quadratic terms around the same solution. Then, in Section 3, to solve the linear-quadratic subproblem, we will propose a Riccati equation method, which will be slightly more general than the standard one. To simplify the argument there, we suppose that the cost function J is a quadratic function with the form given in (11). It is quite enough for our practical purpose. In Section 4, we will state our iterative algorithm (SLQ) in detail. In Section 5, we will propose a game-theoretic controller which automatically adjusts the SLQ method to its enemy's unexpected movements (i.e., different movements from the Nash solution). We will provide results from our numerical experiments for the SLQ method in Section 6 and for the game-theoretic controller in Section 7.

[*] Department of Systems Science and Mathematics, Campus Box 1040, Washington University, One Brookings Drive, St. Louis, MO 63130 USA. E-mail: mukai@zach.wustl.edu, Telephone: (314)935-6064, Fax: (314)935-6121. Effort sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-99-2-0551. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

[†] On leave from Department of Mechanical and System Engineering, Kyoto Institute of Technology, Matsugasaki, Sakyo, Kyoto, 606-8585, Japan.

2. Sequential Linear-Quadratic Method

We propose a numerical method for solving the game problem (3). We assume that an approximate solution u_i is available and we try to improve it. Let $u_i = \left(u_i^B, u_i^R\right)$ be the i-th approximate solution and $x_i = x[u_i]$ be the trajectory corresponding to control u_i with $x_i(t_0) = z_0$. Let $\delta u = \left(\delta u^B, \delta u^R\right)$ be a small perturbation of $u = \left(u^B, u^R\right)$.

Expanding the differential equation (1) around (u_i, x_i), we obtain the following linear approximation to the differential equation (1):

$$\frac{d}{dt}\delta x(t) = f_x(x_i(t), u_i(t))\delta x(t) \\ +f_u(x_i(t), u_i(t))\delta u(t), \;\; \delta x(t_0) = 0, \qquad (4)$$

where $f_x(x_i, u_i) = \frac{\partial f}{\partial x}(x_i, u_i)$ and $f_u(x_i, u_i) = \frac{\partial f}{\partial u}(x_i, u_i)$. Thus, for problem (3), we propose an iterative process, whose i-th step consists of solving the following subproblem, in which the original differential equation (1) is replaced by its linear approximation (4) and the original cost function J is approximated by its quadratic expansion around (x_i, u_i):

$$\min_{\delta u^B} \max_{\delta u^R} \left\{ \int_{t_0}^{t_f} \left[g(x_i, u_i) + g_x(x_i, u_i)\delta x + g_u(x_i, u_i)\delta u \right.\right. \\ +\frac{1}{2}\delta x' g_{xx}(x_i, u_i)\delta x + \frac{1}{2}\delta u' g_{ux}(x_i, u_i)\delta x \\ \left. +\frac{1}{2}\delta x' g_{xu}(x_i, u_i)\delta u + \frac{1}{2}\delta u' g_{uu}(x_i, u_i)\delta u \right] dt \\ +g_f(x_i(t_f)) + \left(g_f\right)_x (x_i(t_f))\delta x(t_f) \\ +\frac{1}{2}\delta x(t_f)' \left(g_f\right)_{xx} (x_i(t_f))\delta x(t_f) \;\Big| \\ \left. \frac{d}{dt}\delta x = f_x(x_i, u_i)\delta x + f_u(x_i, u_i)\delta u, \delta x(t_0) = 0 \right\}, \qquad (5)$$

where, in the interest of compactness, the time t is suppressed. Here, we denote transposition by a prime and the second-order partial derivatives of the function $g(x, u)$ by $g_{xx}(x, u), g_{xu}(x, u), g_{ux}(x, u)$, and $g_{uu}(x, u)$.

Since the subproblem (5) is a linear-quadratic problem with respect to δu and δx, we can employ a Riccati equation method, which will be stated in the next section. We update the approximate solution u_i by

$$u_{i+1} = u_i + \alpha_i \, \delta u_i \qquad (6)$$

with a step size $\alpha_i > 0$, where $\delta u_i = (\delta u_i^B, \delta u_i^R)$ denotes the Nash solution to problem (5). Then we update x_i by

$$x_{i+1} = x[u_{i+1}]. \qquad (7)$$

3. Linear-Quadratic Games

The value function $I(t, z)$ is defined to be the (optimal) value of the game problem (3) when the initial time t_0 is replaced by $t \in [t_0, t_f]$ and the initial state z_0 is replaced by $z \in \mathbb{R}^n$. Under the assumption of continuous differentiability, a direct application of the principle of optimality to $I(t, z)$ yields the so-called Hamilton-Jacobi-Isaacs (HJI) equation,

$$-I_t(t, z) = \min_{u^B} \max_{u^R} \left[I_z(t, z) f(z, u, t) + g(z, u, t)\right], \quad (8)$$

with boundary condition

$$I(t_f, z) = g_f(t_f, z) \qquad \text{for any } z \in \mathbb{R}^n. \qquad (9)$$

If there exists a differentiable function $I(t, z)$ satisfying (8) and (9), then the HJI equation provides a means for obtaining a Nash solution.

For this section, we suppose that the cost function J is a quadratic function and we consider the following affine-quadratic problem:

$$\min_{u^B} \max_{u^R} \left\{ \tilde{J}(x; u^B, u^R) \;\Big|\; \frac{d}{dt}x(t) = A(t)x(t) \right. \\ \left. +B^B(t)u^B(t) + B^R(t)u^R(t) + c(t), \; x(t_0) = z_0 \right\}, \quad (10)$$

where

$$\tilde{J}(x; u^B, u^R) = \frac{1}{2} \int_{t_0}^{t_f} \left[x(t)'Q(t)x(t) + 2x(t)'d(t) \right. \\ +u^B(t)'R^B(t)u^B(t) + 2u^B(t)'r^B(t) \\ \left. -u^R(t)'R^R(t)u^R(t) - 2u^R(t)'r^R(t) \right] dt \\ +\frac{1}{2}x(t_f)'Q_f x(t_f) + x(t_f)'r_f. \qquad (11)$$

Here, the square matrices $Q(t), R^B(t), R^R(t)$ and $Q_f(t)$ are symmetric. The matrices $R^B(t)$ and $R^R(t)$ are assumed to be positive definite, while $Q(t)$ and $Q_f(t)$ are positive semi-definite. As in [1] and [2], we can expect that the value function $I(t, z)$ is a quadratic function of z:

$$I(t, z) = \frac{1}{2}z'S(t)z + k(t)'z + m(t), \qquad (12)$$

where $S(t) \in \mathbb{R}^{n \times n}$, $k(t) \in \mathbb{R}^n$ and $m(t) \in \mathbb{R}^n$. For this case we can solve equation (8) explicitly (see [2]).

Lemma 1 (Riccati equations) The Hamilton-Jacobi-Isaacs equation (8) for the linear-quadratic problem (10) has a solution $I(t, z)$ of the form (12) on $[t_0, t_f] \times \mathbb{R}^n$ if the following system of equations has a solution (S, k, m):

$$\frac{d}{dt}S(t) + S(t)A(t) + A(t)'S(t)$$

$$
\begin{aligned}
&-S(t)B^B(t)R^{B^{-1}}(t)B^B(t)'S(t)\\
&+S(t)B^R(t)R^{R^{-1}}(t)B^R(t)'S(t)+Q(t)=0, \qquad (13)\\
&\frac{d}{dt}k(t)+A(t)'k(t)-S(t)B^B(t)R^{B^{-1}}(t)B^B(t)'k(t)\\
&+S(t)B^R(t)R^{R^{-1}}(t)B^R(t)'k(t)\\
&-S(t)B^B(t)R^{B^{-1}}(t)r^B(t)-S(t)B^R(t)R^{R^{-1}}(t)r^R(t)\\
&+S(t)c(t)+d(t)=0, \qquad (14)\\
&\frac{d}{dt}m(t)-\frac{1}{2}k(t)'B^B(t)R^{B^{-1}}(t)B^B(t)'k(t)\\
&+\frac{1}{2}k(t)'B^R(t)R^{R^{-1}}(t)B^R(t)'k(t)\\
&-k(t)'B^B(t)R^{B^{-1}}(t)r^B(t)-k(t)'B^R(t)R^{R^{-1}}(t)r^R(t)\\
&-\frac{1}{2}r^B(t)'R^{B^{-1}}(t)r^B(t)+\frac{1}{2}r^R(t)'R^{R^{-1}}(t)r^R(t)\\
&+k(t)'c(t)=0, \qquad (15)
\end{aligned}
$$

with the terminal conditions,

$$S(t_f)=Q_f,\ \ k(t_f)=r_f,\ \ m(t_f)=0. \qquad (16)$$

We can obtain the following explicit formula for the Nash control in a state feedback form.

Proposition 2 Suppose that a solution (S,k,m) to the equations (13)-(15) with (16) exists on all of $[t_0,t_f]$. Then a Nash solution u^* to the affine-quadratic differential game (10) is found from

$$
\begin{aligned}
(u^B)^*(t) &= -R^{B^{-1}}(t)\{B^B(t)'(S(t)x^*(t)+k(t))\\
&\qquad +r^B(t)\}, \qquad (17)\\
(u^R)^*(t) &= R^{R^{-1}}(t)\{B^R(t)'(S(t)x^*(t)+k(t))\\
&\qquad -r^R(t)\}, \qquad (18)
\end{aligned}
$$

and the corresponding value is given by

$$\tilde{J}(x^*;u^*)=\frac{1}{2}z_0'S(t_0)z_0+k(t_0)'z_0+m(t_0), \qquad (19)$$

where $x^*=x[u^*]$ is the state trajectory driven by the control u^*. □

Then, just like the well known standard form of the Riccati Equation Method, we may compute the Nash solution u^* by the following procedure: Substituting (17) and (18) into the linear ordinary equation in (10), we obtain x^* as the solution to the initial value problem

$$
\begin{aligned}
&\frac{d}{dt}x(t)=[A(t)-B^B(t)R^{B^{-1}}(t)B^B(t)'S(t)\\
&+B^R(t)R^{R^{-1}}(t)B^R(t)'S(t)]x(t)\\
&-B^B(t)R^{B^{-1}}(t)\{B^B(t)'k(t)+r^B(t)\}\\
&+B^R(t)R^{R^{-1}}(t)\{B^R(t)'k(t)-r^R(t)\},x(t_0)=z_0.(20)
\end{aligned}
$$

Finally, we compute the optimal control u^* by (17) and (18).

4. SLQ Iterative Algorithm

We first note that the coefficients in (10) and (11) are defined as follows:

$$
\begin{aligned}
&A(t)=f_x(x_i(t),u_i(t)),\ B^B(t)=f_{u^B}(x_i(t),u_i(t)),\\
&B^R(t)=f_{u^R}(x_i(t),u_i(t)),\quad c(t)=0,\\
&d(t)=g_x(x_i(t),u_i(t)),\ r^B(t)=g_{u^B}(x_i(t),u_i(t)),\\
&r^R(t)=-g_{u^R}(x_i(t),u_i(t)),\ Q(t)=g_{xx}(x_i(t),u_i(t)),\\
&R^B(t)=g_{u^Bu^B}(x_i(t),u_i(t)),\\
&R^R(t)=-g_{u^Ru^R}(x_i(t),u_i(t)).
\end{aligned}
$$

The sequential linear-quadratic algorithm SCL thus has the following form:

Sequential Linear-Quadratic Algorithm

Step 0: Select a stopping criterion $\varepsilon>0$, and an initial control-trajectory pair (u_0,x_0) with $x_0=x[u_0]$. Set the counter $i=0$.

Step 1: Solve the Riccati equations, (13), (14), (15) and (16), and obtain the solution $(S_i(t),k_i(t),m_i(t))$.

Step 2: Solve the linear ordinary differential equation (20) from the initial state $z_0=0$ and rename its solution $\delta x_i(t)$.

Step 3: Compute $\delta u_i(t)$ using (17) and (18), where $x^*(t)$ stands for $\delta x_i(t)$ and $(u^B)^*(t)$ and $(u^R)^*(t)$ respectively stand for $\delta u_i{}^B(t)$ and $\delta u_i{}^R(t)$.

Step 4: Set $u_{i+1}(t)=u_i(t)+\alpha_i\,\delta u_i(t)$ with a step size $\alpha_i>0$.

Step 5: Compute a new trajectory x_{i+1} by solving the original ordinary differential equation (1) with $u(t)$ replaced by $u_{i+1}(t)$.

Step 6: If $\sup\{||\delta u_i(t)||: t_0\le t\le t_f\}\le\varepsilon$, stop; otherwise, go to Step 1 with i replaced by $i+1$.

5. Game-Theoretic Controller

We present a controller for one force, the Blue, in which we automatically adjust the SLQ method to its enemy's unexpected movements (i.e., different movements from the Nash solution by the Red). Namely, we consider the situation that the Red force chooses an arbitrary control $u^R(t)=u^{*R}(t)+\delta u^R(t)$ instead of the Nash solution $u^{*R}(t)$. Then, the Blue force may want to choose a control $u^B(t)=u^{*B}(t)+\delta u^B(t)$ where an additional term $\delta u^B(t)$ should works against the unexpected part of the Red force's control. By considering the feedback law (17) for small

$\delta u(t)$ and $\delta x(t)$, the Blue force should choose the control of the form

$$u^B(t) = u^{*B}(t) + K^B(t)\delta x(t). \tag{21}$$

Namely, the Blue force may want to use the controller output $\delta u^B(t)$ of the above form which contains the current state $x(t)$ as a controller input besides the Nash solution $u^{*B}(t)$ and the feedback law (17). Thus, we propose the following method.

Game-Theoretic Controller Simulation

Step 0: Select a state deviation tolerance $\gamma > 0$. Set $\tau = t_0$, $z_0 = x(t_0)$ and the counter $i = 0$.

Step 1: Solve the game problem over $[\tau, t_f]$

$$\min_{u^B} \max_{u^R} \Big\{ J[x; u^B, u^R] \;\Big|\; \frac{d}{dt}x(t) = f(x(t), u(t)),$$
$$x(\tau) = z_0 \Big\} \tag{22}$$

and compute the Nash solution (x^*, u^*) and the feedback gain $K^B(t)$ over $[\tau, t_f]$.

Step 2: Solve the nonlinear differential equation

$$\frac{d}{dt}x(t) = f\Big(x(t), u^{*B}(t) + K^B(t)\big(x(t) - x^*(t)\big),$$
$$u^{*R}(t) + \delta u^R(t)\Big) \tag{23}$$

on $[\tau, t_f]$ with the initial condition $x(\tau) = z_0$.

Step 3: Compute $s = \inf\{t \in [\tau, t_f] : \|\delta x(t)\| \geq \gamma\}$, where $\delta x(t) \equiv x(t) - x^*(t)$ on $[\tau, s]$.

Step 4: Set $u^B(t) = u^{*B}(t) + K^B(t)\big(x(t) - x^*(t)\big)$ on $[\tau, s]$. Use this control $u^B(t)$ on $[\tau, s]$.

Step 5: If $s = t_f$, stop; otherwise, set $z_0 = x(s)$ and $\tau = s$, and go to Step 1 with i replaced by $i + 1$.

6. Numerical Experiments: Performance of the SLQ Method

In this section, we apply the SLQ method to a dynamic model of air operations for the military, and report on its numerical results. We consider a differential game between two opposing forces, the Blue and the Red, each of which has (generally) multiple units and is operating in an geographical area, a theater of operations. Our model is represented by a nonlinear ordinary differential equation (1), whose state consists of the position of each unit in $\mathbb{R}^2$, the number of platforms (e.g., bombers, fighter-interceptors, SAM missile launchers and so on) in each unit, and the number of weapons carried by each platform. Each unit has the control inputs consisting of the velocity for its position, and the firing intensity in its engagement with its enemy.

6.1. One Blue versus One Red

In this subsection, we consider the simplest case: Each of the Blue and Red forces has one unit. Both the Blue unit (**B1**) and the Red unit (**R1**) have 10 platforms. Since each unit has a 4-dimensional state and a 3-dimensional control input, the entire model has a 8-dimensional state and a 6-dimensional control input. The unit movement dynamics and the platform attrition dynamics (i.e., the specific forms of system equation (1) and cost function (11)) will be given in other reports [3].

In this experiment, each force has two objectives: i) to reach its specified fixed target; and ii) to reduce the number of enemy platforms while preserving the number of its own. The initial positions are given by the following coordinates relative to a theater of operations of size 100 by 100: $(20, 50)$ for **B1** and $(50, 80)$ for **R1**. The location of targets are given by the following: $(80, 50)$ for **B1** and $(50, 20)$ for **R1** (see Figure 2).

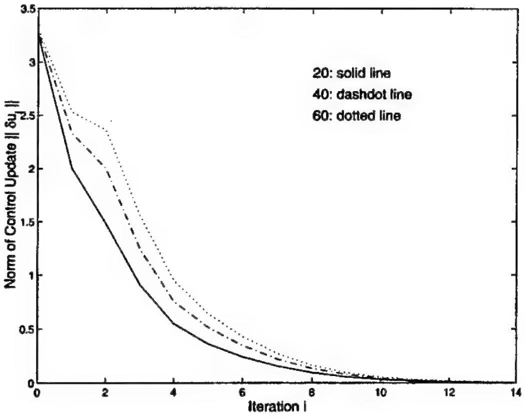


Figure 1. Convergence of the SLQ Method

We considered the situation that the Blue unit **B1** is less concerned about its own survival but the Red unit **R1** is more concerned about it. For instance, the Blue unit consists of interceptors whose aim is to shoot down red targets consisting of bombers. The bombers on the other hand have as primary objective to reach their target and will avoid contact en route. So we put a lower weight on the terminal state of the Blue's platforms and put a much higher weight on the terminal state of the Red's platforms. Actually, we gave three different weights, 20, 40 and 60, for the terminal state of the Red's platforms and gave 0.2 for the weight on the terminal state of the Blue's platforms. As shown in Figure 2, when the two units get close, the Red unit **R1** tries to escape and avoid being shot by the Blue unit **B1**. On the other hand, the Blue unit **B1** tries to pursue the Red unit **R1** and fire at it with almost its maximum firing intensity as seen from Figures 2 and 3. In Figures 1-5, the solid line

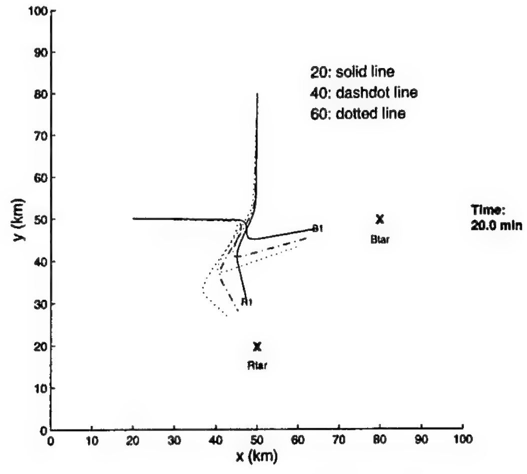


Figure 2. Nash trajectories

stands for the weight of 20, the dash-dot line for the weight of 40 and the dotted line for the weight of 60. We can easily observe that such a pursuit evasion game becomes even more obvious when we increase the weight on the terminal state of the Red's platforms.

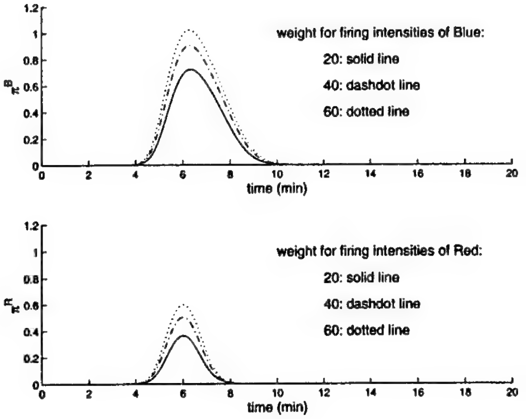


Figure 3. Nash firing intensities

The method converged to a solution in 14 iterations starting from a nominal solution. Figure 1 shows the norm of the control correction δu_i versus iteration i. Figures 2-4 show the solution. Figure 2 shows the movements of the 2 units in the theater over a time period of 20 minutes. After an engagement in the middle, the units head for their respective fixed targets. Figure 3 shows the firing intensity control as a function of time. The firing intensity of each unit increases when its target unit is near by. Figure 4 shows how the number of platforms goes down for each unit. The Red unit of bombers (**R1**) suffers heavy casualties. Figure 5 shows how the number of weapons goes down for each unit.

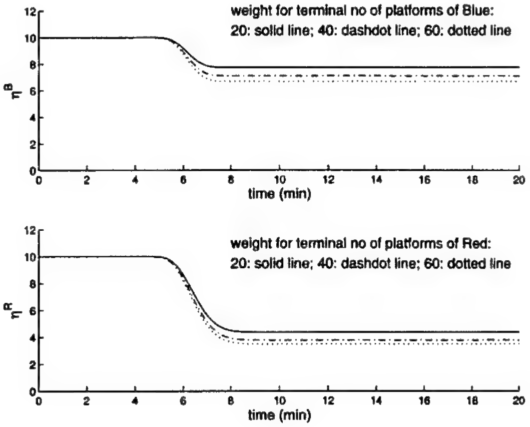


Figure 4. Nash numbers of platforms

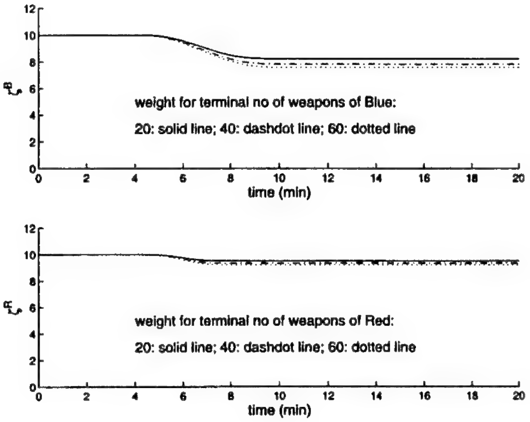


Figure 5. Nash numbers of weapons

6.2. Two Blue versus Two Red

In this subsection, we consider the following specific problem: Each of the Blue and Red forces has two units. The first Blue unit (**B1**) consists of 10 bombers as platforms, and the second Blue unit (**B2**) consists of 10 fighters. On the other hand, the first Red unit (**R1**) consists of 10 bombers, and the second Red unit (**R2**) consists of 10 interceptors. Since each unit has a 4-dimensional state and a 3-dimensional control input, the entire model now has a 16-dimensional state and a 12-dimensional control input. The unit movement dynamics and the platform attrition dynamics (i.e., the specific forms of system equation (1) and cost function (11)) again will be given in [3].

In this experiment, each unit on either force has two objectives: i) to reach its specified fixed target; and ii) to reduce the number of enemy platforms while preserving the number of its own. The initial positions are given by the following: (20, 10) for **B1**, (20, 50) for **B2**, (80, 12) for **R1** and (80, 52) for **R2**. The location of targets are given by the following: $(80, 10)$ for **B1**, $(80, 50)$ for **B2**, $(20, 12)$ for **R1** and $(20, 52)$ for **R2**.

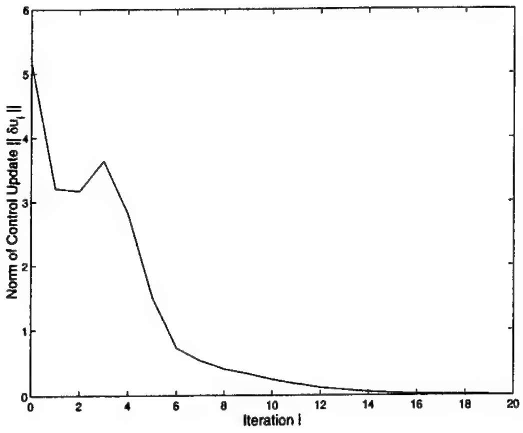


Figure 6. Convergence of the SLQ Method

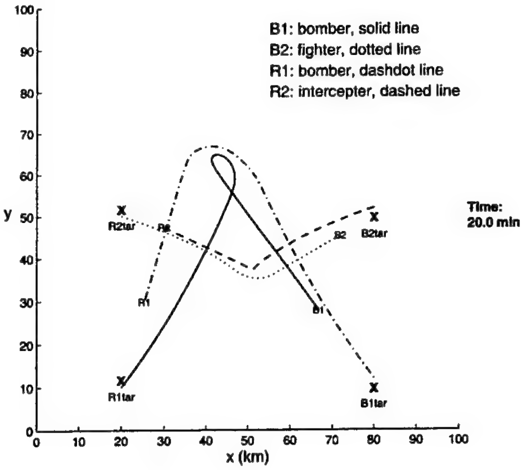


Figure 7. Nash trajectories

We have introduced an asymmetry in the relative weights for the number of platforms. For **B1** and **R2**, the weights are selected as 0.1, while for **B2** it is 10 and for **R1** it is 40. We would expect that **B1** and **R2** would be less concerned about their own survival, and therefore pursue and attack an enemy unit. On the other hand, **B2** and **R1** should be expected to evade from their pursuers. As shown in Fig. 7, the behavior of the units for the Nash equilibrium confirm these expectations.

The method converged to a solution in 20 iterations starting from a nominal solution. Figure 6 shows the norm of the control correction δu_i versus iteration i. Figures 7-10 show the solution. Figure 7 shows the movements of the 4 units in the theater over a time period of 20 minutes. After an engagement in the middle, the units head for their respective fixed targets.

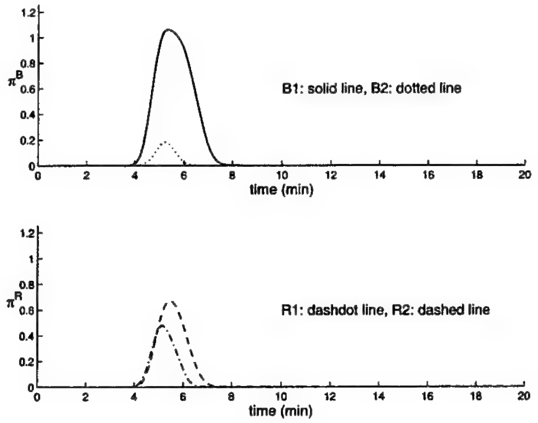


Figure 8. Nash firing intensities

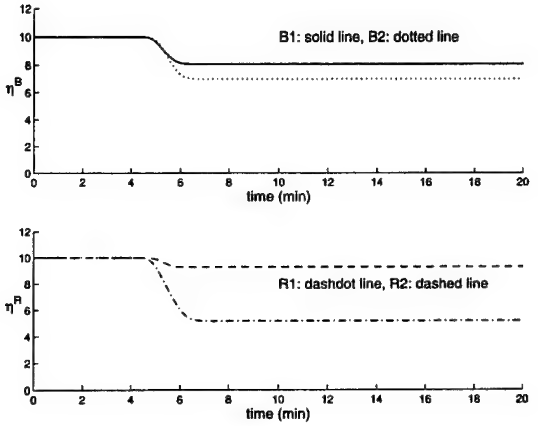


Figure 9. Nash numbers of platforms

Figure 8 shows the firing intensity control as a function of time. The firing intensity of each unit increases when its target unit is near by. Figure 9 shows how the number of platforms goes down for each unit. The Red unit of bombers (**R1**) suffers heavy casualties. Figure 10 shows how the number of weapons goes down for each unit.

7. Numerical Experiments: Performance of the Game-Theoretic Controller under Noisy Observation and Model Mismatch

In this section, we apply the game-theoretic controller to a dynamic model of air operations for the military, and discuss its performance under noisy observation and model mismatch. We report three experiments.

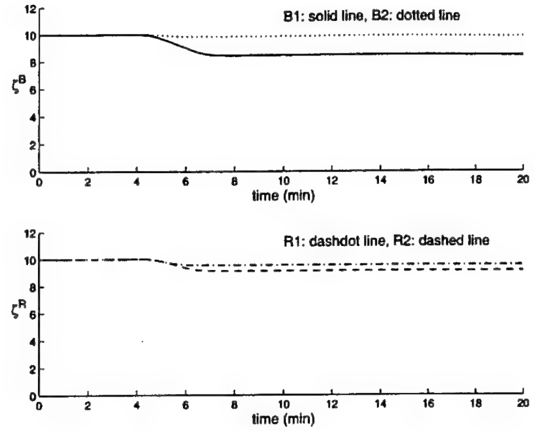


Figure 10. Nash numbers of weapons

7.1. Experiment 1

The set-up of experiment 1 is as in Section 6.1 and we use identical weights on the distances to their fixed targets for the Red and the Blue units, as well as the same weights on velocity or firing intensity commands. The weights on the number of platforms are different, namely 0.01 and 3.00 respectively for the Blue and the Red unit.

In this experiment, we observe how the performance of the controller deteriorates as the standard deviation of the observation noise is increased. We add white noise with zero mean to the Red unit's position and observe how the cost (in Figure 11), the respective numbers of platforms (in Figure 12) and the respective distances to targets (in Figure 13) at final time change. Thus, while the Blue unit's information is corrupted by noise, we still assume perfect information for the Red unit. For each value of standard deviation, we run 200 sample paths and compute the respective expected values.

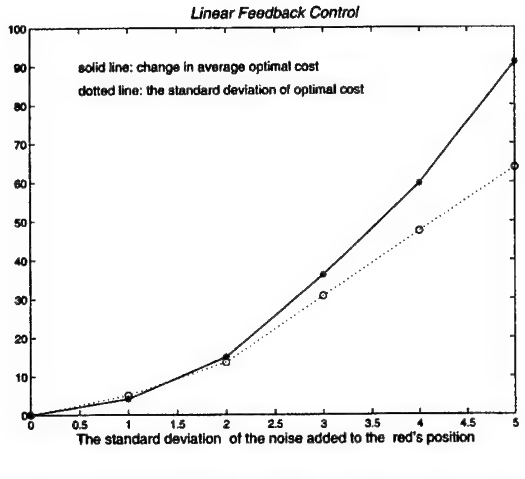


Figure 11. Changes in cost with noise

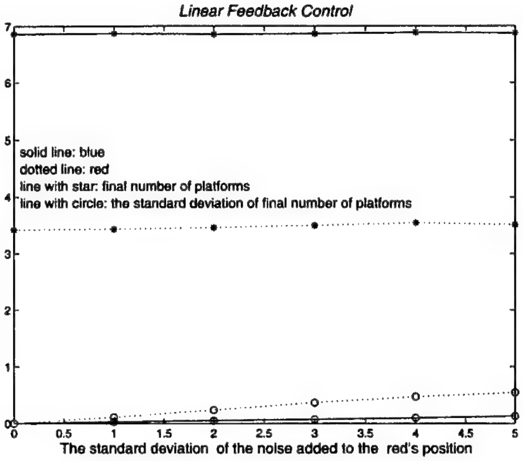


Figure 12. Numbers of platforms with noise

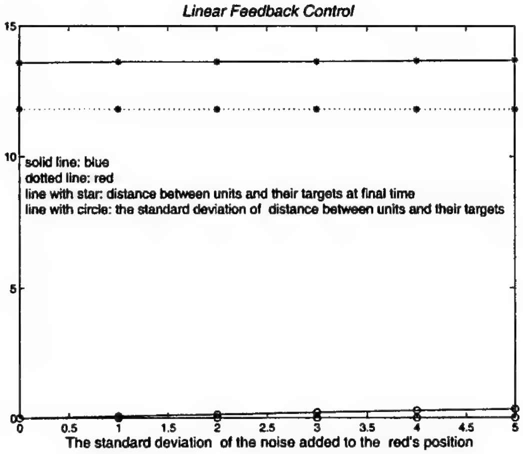


Figure 13. Distance to target with noise

In Figure 11, as the standard deviation of the noise increases, the optimum cost value increases as well. The objective of the Blue force is to reduce the cost. Hence the Blue force's performance deteriorates as noise in the Blue's observation of the Red's position increases. The total cost is around 4000. Hence the change in the cost is about 2 percent (90/4000) for noise of size 5 kilometers. The optimum cost is therefore not too sensitive to noise in observations.

In Figure 12, the final numbers of platforms also are not very sensitive to the noise. However, as the standard deviation reaches higher values, the Red side does slightly better. This can be explained again by the fact that the Blue side is not getting good information. The Blue side is practically insensitive to the noise.

In Figure 13, the noise does not affect the Red's final distance to its target. The Blue's final distance to its target changes only slightly, as the standard deviation of the noise increases.

Summarizing: The optimum cost is only slightly sensitive to noise in observation of the Red unit's position. The numbers of platforms and the respective distances to the final targets are even less sensitive. Further investigations will include an experimental set-up in which two controllers for the Blue and the Red units are separated from the plant and both parties will receive incomplete information.

7.2. Experiment 2

In this experiment, we consider the case in which the internal model of the plant inside the controller is different from the actual plant model in the sense that the value of one parameter differs. As the non-identical parameter we have selected the probability that the Red unit kills the Blue unit. We observe the performance of the controller as the value of the probability of kill deviates from its true value. We employ both the linear feedback control and the nonlinear feedback control, where we make use of the following simplified terminologies on feedback for our convenience:

- Linear feedback: State feedback around the Nash solution (the control is a linear function of the state),
- Nonlinear Feedback: Once the state deviates sufficiently from the Nash solution, a new Nash solution is computed and linear feedback is used around the computed Nash solution until it also deviates too much.

In this experiment, we observe how the cost (in Figures 14, 15), the respective numbers of platforms (in Figures 16, 17) and the respective distances to targets (in Figures 18, 19) at final time change for both the linear and the nonlinear feedback controls as the value of the probability of kill deviates from its true value.

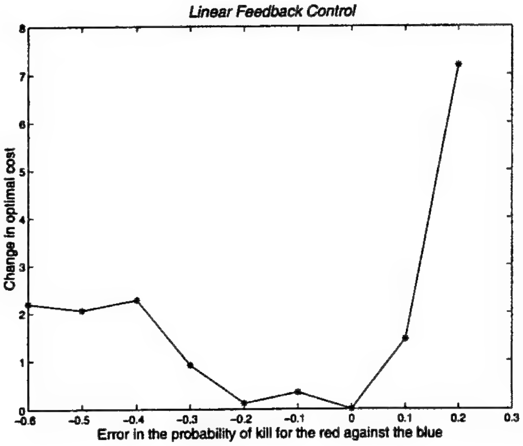


Figure 14. Cost under model mismatch: linear feedback

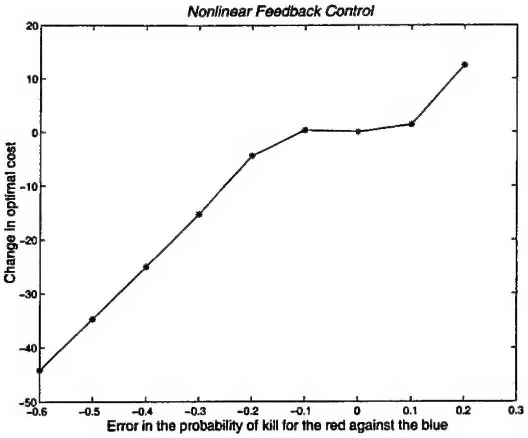


Figure 15. Cost under model mismatch: nonlinear feedback

In Figures 14-15, as the probability of kill varies from 0.2 to 1.0 (or equivalently the error from the true value of 0.8 varies from -0.6 to $+0.2$), the change in the optimum cost is about 0.1 percent for linear feedback control and about 1 percent for nonlinear feedback control respectively. The optimum cost is therefore insensitive to model mismatch in this parameters.

In Figures 16-17, like the previous results, there is no significant change in the final numbers of platforms as different incorrect values of the probability of kill are employed. The final numbers of platforms is insensitive to this kind of model mismatch.

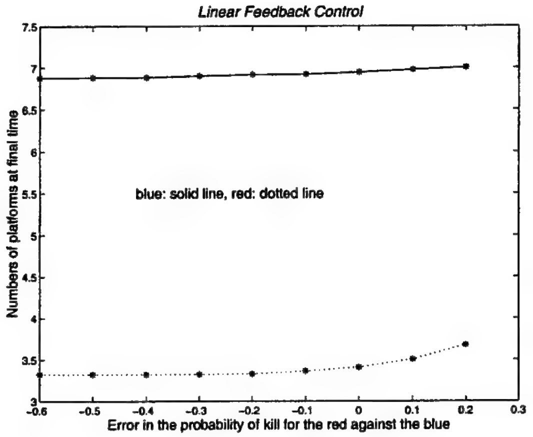


Figure 16. Number of platforms under model mismatch: linear feedback

In Figure 18, as the Blue underestimates the Red's capability (the error in the Red's probability of kill is -0.6),

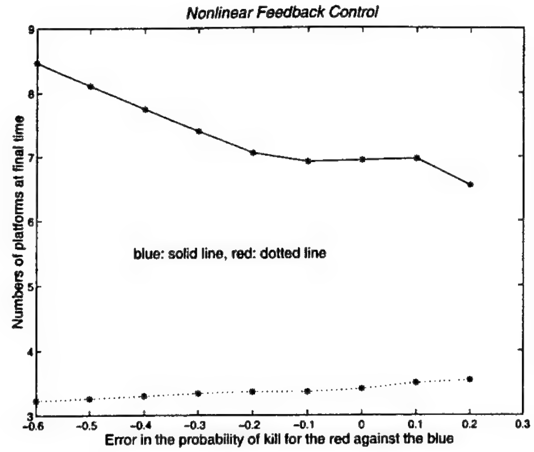


Figure 17. Number of platforms under model mismatch: nonlinear feedback

the Blue's final distance to the target increases. In this manner, the Blue ends up closer to the final target with linear feedback. The final distances to the targets are relatively insensitive to model mismatch.

In Figure 19, similar remarks can be made, for the case of nonlinear feedback. The Blue ends up closer to its final target as the mismatch increases. The final distances to the targets are relatively insensitive to model mismatch.

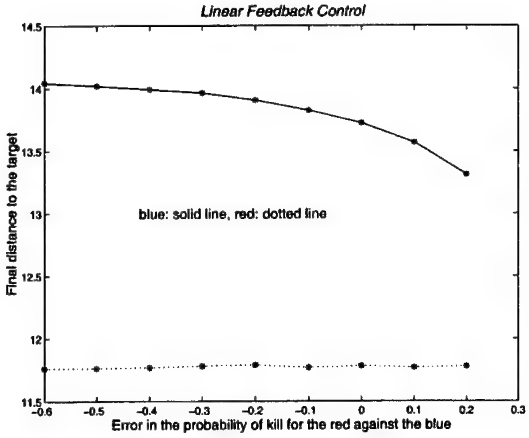


Figure 18. Distance to target under model mismatch: linear feedback

The main conclusion is that, except for the final position, the optimum cost and the final numbers of platforms are relatively insensitive to parametric uncertainty in the probability of kill in the case of linear feedback but are slightly more sensitive in the case of nonlinear feedback. The Blue's final position changes for both linear and nonlinear cases but, in the nonlinear case, Red's final position is affected as well. These all point to the fact that the nonlinear controller is more realistic.

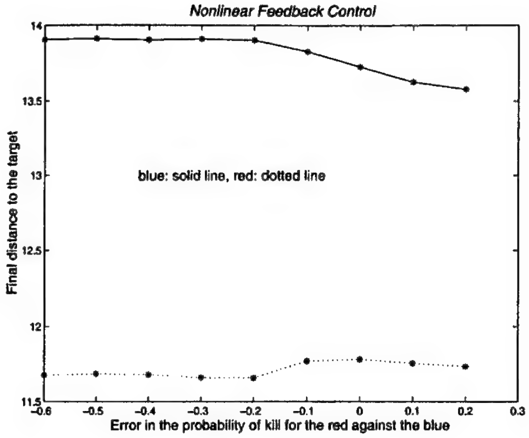


Figure 19. Distance to target under model mismatch: nonlinear feedback

7.3. Experiment 3

In this experiment, we assume that the Blue unit does not know the exact final destination of its target: the Red unit. We use the error step size of 0.1 kilometer in the first coordinate of the position vector.

The changes in the optimum cost and the final numbers of platforms again are relatively insensitive (no figures are included).

The following figures (Figures 20-23) show the optimum trajectories of the units, the change in the optimum cost and the final numbers of platforms for different values of the position error. Clearly the optimum trajectories are quite sensitive to the errors in the reading of the final destination of the enemy, although the optimum cost and the final numbers of platforms are not. The penalty for the final destination error has a large value in the beginning, because the units are farther away from their final destination in the beginning. This is probably the reason behind the sensitivity of the optimum trajectories.

References

[1] B. Anderson and J. B. Moore, *Optimal Control*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[2] T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory*, second edition, Academic Press, New York, New York, 1995.

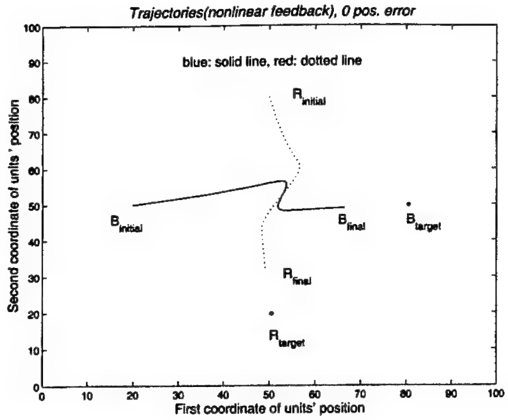


Figure 20. Optimum trajectories with accurate target information

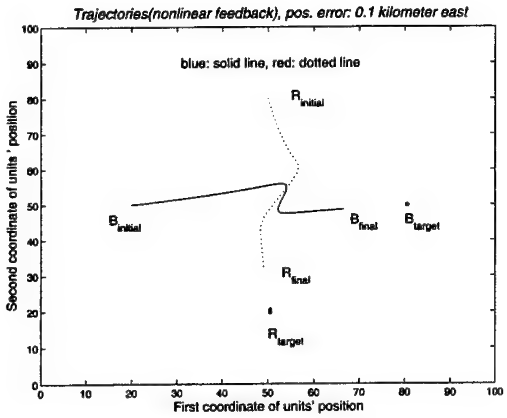


Figure 21. Optimum trajectories with inaccurate target information 1

[3] H. Mukai, Y. Sawada and I. Tunay, ”Mission dynamics continuous-time model”, preprint, 2000.

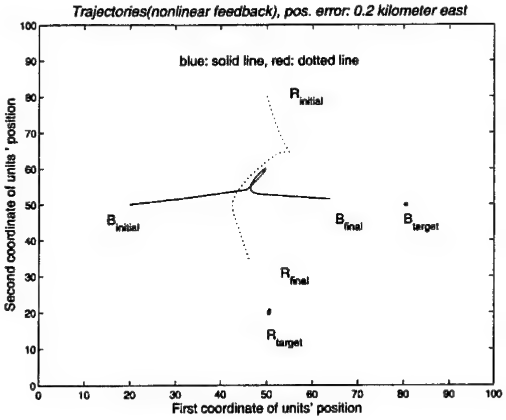


Figure 22. Optimum trajectories with inaccurate target information 2

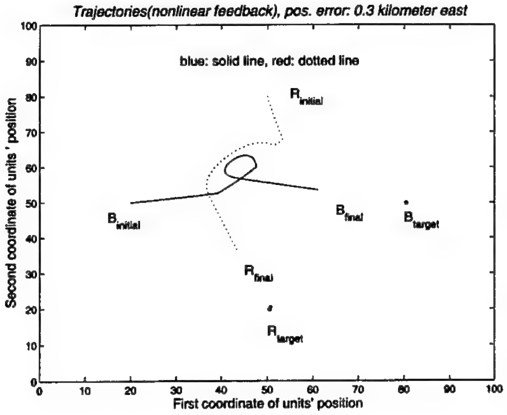


Figure 23. Optimum trajectories with inaccurate target information 3

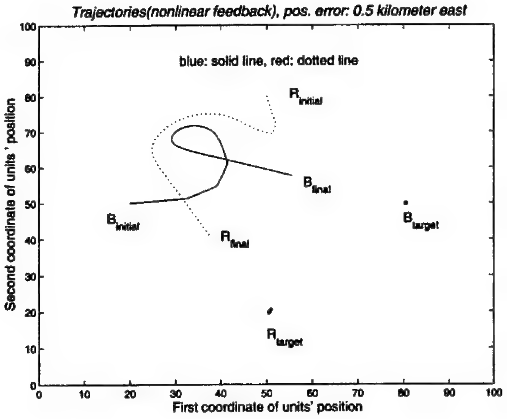


Figure 24. Optimum trajectories with inaccurate target information 4

A State Space Model with Adversarial Control for Military Operations

Jose B. Cruz, Jr[1]., Marwan A. Simaan[2], Aca Gacic[2], Huihui Jiang[1],
Bruno Letellier[1], and Ming Li[1]

[1] *Department of Electrical Engineering*
The Ohio State University,
Columbus, OH 43210
cruz+@osu.edu
(614) 292-1588

[2] *Department of Electrical Engineering*
University of Pittsburgh
Pittsburgh, PA 15261
simaan@ee.pitt.edu
(412) 624-8099

Abstract

In this paper we present a nonlinear state space mathematical model for a class of dynamical systems that can serve as the basis for a simulation test bed for the investigation of enterprise control. Dynamic complex enterprises generally include multiple control agents of a decision team. In addition, the enterprise is generally imbedded in a larger environment that has competing and even hostile decision teams that affect the enterprise. In such situations it is appropriate to model an extended enterprise that includes the competing decision teams. For example an enterprise might be a military command and control hierarchy with several levels of command. If a command and control enterprise is deployed in a military operation, the enterprise states may be affected by non-friendly commands. In order to develop acceptable and even optimal control strategies, it is important to consider the effect of the adversarial controls even at the control design stage. Before these control strategies can be designed or investigated, a model for the extended enterprise "plant" is needed. This extended plant should have inputs from the competing decision team, in addition to the decision team inputs to the enterprise. In order to gain concrete insights to enterprise control, a class of enterprises, command and control, is chosen. The command hierarchy will be designated as the Blue Forces. The enterprise is imbedded in a larger system that includes a hostile command designated as the Red Forces. This extended enterprise will be designated as "Military Operations". In this paper, a discrete-time nonlinear state space model of the Military Operation system is presented.

1. Introduction

Attrition models for modern warfare have received considerable attention in recent years [1-4]. In this paper, we present a dynamic state-space attrition-type model of a complex military operation that involves two opposing forces. We will label the attacking forces a *Blue* and the defending forces as *Red*. The blue forces consist of Blue Weasels (BWs) and Blue Bombers (BBs). The weasels are essentially SEAD[1] units whose purpose is to attack and suppress the red air defenses, and the purpose of the bombers is to attack the red units. The red forces consist of Red Troops (RTs) such as tanks and mobile vehicles and Red Defense units (RDs) such as SAM's[2]. In addition, we will assume that there are Fixed Targets (FTs) such as bridges, refineries, air bases, etc. that the blue forces would attack and the red forces would defend.

Let $N^{BW}, N^{BB}, N^{RT}, N^{RD}$, and N^{FT} denote the number of units of each type involved in the operation. Although the model can be derived in the continuous time-space domain, we will initially assume that time is sampled into stages $k = 0,1,2,...K$ and that the scenario is taking place on a two-dimensional terrain sampled in the x-y directions into square blocks. Continuous time and three-dimensional continuous space will be considered as an extension of this work at a later time.

[1] SEAD = Suppressing Enemy Air Defenses
[2] SAMs = Surface to Air Missiles

2. The Unit's State Vector

Consider the i^{th} unit of type X where $X = \{BW, BB, RT, RD\}$. Let $\xi_i^X(k) = \begin{bmatrix} x_i^X(k) \\ y_i^X(k) \end{bmatrix}$ denote its location vector at time k, where x is the horizontal coordinate and y is the vertical coordinate. Let $p_i^X(k)$ denote the number of platforms and let $w_i^X(k)$ denote the average number of weapons per platform at time k in that unit. Thus, for each moving unit in the theatre of operations, we will define a 4-dimensional state variable

$$z_i^X(k) = \begin{bmatrix} \xi_i^X(k) \\ p_i^X(k) \\ w_i^X(k) \end{bmatrix}, \qquad X = \{BW, BB, RT, RD\},$$

$$i = 1, 2, \ldots\ldots, N^X, \;\; k = 0, 1, 2, 3 \ldots . K$$

Combining all the state variables for each type of forces into one vector, we can write:

$$z^X(k) = \begin{bmatrix} z_1^X(k) \\ \cdot \\ \cdot \\ z_{N^X}^X(k) \end{bmatrix}.$$

The overall state vectors corresponding to the Blue and Red forces are defined as:

$$z^B(k) = \begin{bmatrix} z^{BW}(k) \\ z^{BB}(k) \end{bmatrix}, \text{ and } z^R(k) = \begin{bmatrix} z^{RT}(k) \\ z^{RD}(k) \end{bmatrix}$$

Now, for the fixed targets, we will assume that their fixed positions are determined by the vectors $\xi_i^{FT} = \begin{bmatrix} x_i^{FT} \\ y_i^{FT} \end{bmatrix}$, $i = 1, 2, \ldots\ldots FT$. Let $p_i^{FT}(k)$ denote the number of platforms in the i^{th} fixed target at time k. These platforms carry no weapons and are subject to attack by the Blue forces. We can define a state vector for the fixed targets as:

$$z^{FT}(k) = \begin{bmatrix} p_1^{FT}(k) \\ \cdot \\ \cdot \\ p_{N^{FT}}^{FT}(k) \end{bmatrix}, \;\; k = 0, 1, 2, 3 \ldots . K$$

Combining the state vectors for the Blue and Red forces as well as the state vector for the fixed targets, we can define a state vector for the entire operation as:

$$z(k) = \begin{bmatrix} z^B(k) \\ z^R(k) \\ z^{FT}(k) \end{bmatrix}$$

This will be a $4 \times (N^{BW} + N^{BB} + N^{RT} + N^{RD}) + N^{FT}$ dimensional vector.

3. The Command Variables

We will assume that each moving unit has the following command (or control) variables at each time k:

<u>i) Relocate</u>: A unit can decide to relocate (move) to another adjacent point on the grid. The corresponding control command is:

$$r_i^X(k) = \begin{bmatrix} a_i^X(k) \\ b_i^X(k) \end{bmatrix},$$

where $a_i^X(k) \in \{-1, 0, +1\}$ and $b_i^X(k) \in \{-1, 0, +1\}$ and where a corresponds to the move in the x-direction and b corresponds to the move in the y-direction. There are eight neighboring locations that each unit can relocate to. The $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ option corresponds to the unit deciding to remain in its current location.

<u>ii) Fire Control</u>: Each unit has an option to fire or not to fire. When a unit decides to fire, it must decide on the salvo size. There is a finite set of options for salvo size at each time k. Thus the corresponding control is

$$c_i^X(k) \in \{0, 1, 2, 3 \ldots, C_i^X(k)\}$$

where $C_i^X(k)$ is the largest salvo size that can be fired at time k. Note that if a unit decides not to fire, then $c_i^X(k)=0$

iii) Choice of Target: Each unit can fire only at one target of the opposing forces. If $d_i^X(k)$ denotes the choice of target for unit i at time k, then

$$d_i^{BB}(k)=\{RT_j,\ RD_j,\ or FT_j \text{ for some j}\}$$
$$d_i^{BW}(k)=\{RT_j,\ RD_j,\ or FT_j \text{ for some j}\}$$
$$d_i^{RT}(k)=\{BW_j,\ or BB_j \text{ for some j}\}$$
$$d_i^{RD}(k)=\{BW_j,\ or BB_j \text{ for some j}\}$$

Combining all the command variables into one 4-dimensional control vector, we have the following control vector for each unit

$$u_i^X(k)=\begin{bmatrix} a_i^X(k) \\ b_i^X(k) \\ c_i^X(k) \\ d_i^X(k) \end{bmatrix}.$$

We will now define a composite control vector for each type of forces:

$$u^{BW}(k)=\begin{bmatrix} u_1^{BW}(k) \\ u_2^{BW}(k) \\ . \\ . \\ u_{N^{BW}}^{BW}(k) \end{bmatrix}, \text{ and } u^{BB}(k)=\begin{bmatrix} u_1^{BB}(k) \\ u_2^{BB}(k) \\ . \\ . \\ u_{N^{BB}}^{BB}(k) \end{bmatrix}$$

for the Blue units and

$$u^{RT}(k)=\begin{bmatrix} u_1^{RT}(k) \\ u_2^{RT}(k) \\ . \\ . \\ u_{N^{RT}}^{RT}(k) \end{bmatrix}, \text{ and } u^{RD}(k)=\begin{bmatrix} u_1^{RD}(k) \\ u_2^{RD}(k) \\ . \\ . \\ u_{N^{RD}}^{RD}(k) \end{bmatrix}$$

for the Red units. The overall control vectors for the Blue and Red forces can be represented as:

$$u^B(k)=\begin{bmatrix} u^{BW}(k) \\ u^{BB}(k) \end{bmatrix}, \text{ and } u^R(k)=\begin{bmatrix} u^{RT}(k) \\ u^{RD}(k) \end{bmatrix}.$$

The dimensionality of these vectors will be $4\times(N^{BW}+N^{BB})$ and $4\times(N^{RT}+N^{RD})$ respectively .

4. The Command Constraints

There are numerous constraints that the above command variables must satisfy:

i) Relocate-Fire constraint. For simplicity, we will assume that a unit cannot relocate and fire at the same time.

ii) Fire-Target constraint: We will assume that that no two units of the same force can fire at the same target of the opposing force.

iii) Salvo size constraint: We will assume that ammunitions are not being replenished during the course of the operation.

5. The Highest Level Commands

We will assume that the blue and red forces each have a highest level of commands. Its purpose is to define:

i) The initial states: The initial positions $\xi_i^X(0)$, numbers of platforms $p_i^X(0)$, and weapons $w_i^X(0)$ for each moving unit.

ii) The corridor: Any constraints on the paths of each unit.

The highest level commands may also be able to provide an incentive for the lower level commands to cooperate as a team [5].

6. The State Equations

The state vector for each moving unit is a 4-dimensional vector consisting of the position subvector ξ_i^X, the number of platforms p_i^X, and the number of weapons per platforms w_i^X in that unit. The state vector for each fixed target is p_i^{FT}. We will now derive equations that relate the state variables at time *k+1* to the state and control variables at time *k*.

i) The position subvectors for all moving units in $X = \{BB, BW, RT, RD\}$ change according to the equation of motions:

$$\xi_i^X(k+1) = \xi_i^X(k) + r_i^X(k)$$

ii) The number of platforms for the moving units changes according to a nonlinear attrition model. For example, for BW,

$$\begin{aligned} p_i^{BW}(k+1) = \quad & p_i^{BW}(k)\Big[1 - \sum_{j=1}^{N^{RT}} Q_{ij}^{BWRT}(k) P_{ij}^{BWRT}(k) \\ & \delta(\xi_i^{BW}(k), \xi_j^{RT}(k))\delta(BW_i, d_j^{RT}(k)) \\ & -\sum_{j=1}^{N^{RD}} Q_{ij}^{BWRD}(k) P_{ij}^{BWRD}(k) \\ & \delta(\xi_i^{BW}(k), \xi_j^{RD}(k))\delta(BW_i, d_j^{RD}(k))\Big] \end{aligned}$$

For other units the equations are similar. In the above expression the terms $Q_{ij}^{XY}(k)$ and $P_{ij}^{XY}(k)$ represents the engagement factor and attrition factor between the attacking unit (j^{th} unit of Y) and the unit being attacked (i^{th} unit of X). These factors are computed from the expressions:

$$Q_{ij}^{XY}(k) \dagger \beta_{p_{ij}}^{XY}(1 \text{✍} e^{\text{✍}\frac{p_j^Y(k)}{p_i^X(k)}})$$

$$P_{ij}^{XY}(k) \dagger \text{⇒✍}(1 \text{✍}\beta_w PK_{ij}^{XY})^{c_j^Y(k)} \text{❢}$$

where $p_i^X(k)$ and $p_j^Y(k)$ are the number of platforms in the i^{th} unit of X and j^{th} unit of Y respectively, $\beta_{p_{ij}}^{XY}$ represents the probability that platform p_j of Y acquires platform p_i of X as a target, β_w represents the weapon probability to acquire the target, PK_{ij}^{XY} represents the probability of kill for a single weapon (i.e., a salvo size of 1) for the type of weapon used by unit j against the type of platform in unit i, and $c_j^Y(k)$ is the salvo size of the weapons fired by the j^{th} unit of *Y* at time *k*. The Kronecker delta, which appears in the above expressions, is defined as

$$\delta(V,W) = \begin{cases} 0 & \text{if } V \neq W \\ 1 & \text{if } V = W \end{cases} .$$

In concise form, the four equations for the number of platforms can be written as:

$$p_i^X(k+1) = f_i^X(z(k), u^B(k), u^R(k), k)$$

iii) The number of weapons per platform for each moving unit changes according to the following expression:

$$\begin{aligned} w_i^{BW}(k+1) = \quad & w_i^{BW}(k) - c_i^{BW}(k) \times \\ & \Big[\sum_{j=1}^{N^{RT}} Q_{ij}^{BWRT}(k)\delta(d_i^{BW}(k), RT_j) \\ & +\sum_{j=1}^{N^{RD}} Q_{ij}^{BWRD}(k)\delta(d_i^{BW}(k), RD_j) \\ & +\sum_{j=1}^{N^{FT}} Q_{ij}^{BWFT}(k)\delta(d_i^{BW}(k), FT_j)\Big] \end{aligned}$$

There are similar equations for the other units. In concise form, these expressions can be written as:

$$w_i^X(k+1) = f_i^X(z(k), u^B(k), u^R(k), k)$$

Combining the state equations for all forces, we get the final expression for the state equation

$$z(k+1) = f(z(k), u^B(k), u^R(k), k)$$

where z is a $4 \times (N^{BS} + N^{BB} + N^{RT} + N^{RD}) + N^{FT}$ dimensional state vector, u^B is an $4 \times (N^{BS} + N^{BB})$ dimensional control vector of the Blue forces and u^R is an $4 \times (N^{RT} + N^{RD})$ dimensional control vector of the Red forces. The function f is a $4 \times (N^{BS} + N^{BB} + N^{RT} + N^{RD}) + N^{FT}$ vector of functions.

7. Illustrative Example

We consider a scenario where the mission of the Blue forces is to attack and destroy an air base that is being defended by Red forces. For simplicity we consider one unit of each of *BW* and *BB* planning the attack and one unit of each of *RD* and *RT* defending the base. Let the grid size over which the attack is taking place consist of 10 x 10 square units of 40 square nautical miles each. The controls and states are being updated every 5 minutes and we consider a run of 24 updates corresponding to a mission duration of 2 hours. The description of forces is as follows:

Fixed Target (FT): An air base with a total of 10 platforms (command center, runways, hangars, etc..) Location: $x_i^{FT} = 2, y_i^{FT} = 2$;

Platform state variable: $p_1^{FT}(k)$;

Initial value: $p_1^{FT}(0) = 10$.

Defending Forces (RED):

Red Defense (RD): One Fixed SAM battery consisting of 6 launchers with 3 fixed SAMs each (SAM-F) and one radar.

Initial Location: $x_1^{RD}(0) = 2$, $y_1^{RD}(0) = 2$;

Platform state variable (launchers + radar): $p_1^{RD}(k)$; Initial value: $p_1^{RD}(0) = 7$;

Weapons state variable (Average # of SAMs per platform): $w_1^{RD}(k)$;

Initial value: $w_1^{RD}(0) = 2.57$.

Red Troops (RT): A mechanized regiment consisting of 3000 soldiers, 200 trucks, 50 armored vehicles and 50 tanks, and equipped with 3 shoulder launched SAMs (SAM-H) per armored vehicle.

Initial Location: $x_1^{RT}(0) = 5, y_1^{RT}(0) = 5$;

Platform state variable (trucks + armored vehicles + tanks): $p_1^{RT}(k)$;

Initial State: $p_1^{RT}(0) = 300$;

Weapons state variable (Average # of SAMs per platform): $w_1^{RT}(k)$;

Initial value: $w_1^{RT}(0) = 0.5$.

Attacking Forces (Blue):

Blue Weasels (BW): Two F2-E fighter planes each equipped with 4 AGM2 (air to ground) missiles.

Initial Location: $x_1^{BW}(0) = 8$, $y_1^{BW}(0) = 6$;

Platform state variable (F2-E fighters): $p_1^{BW}(k)$.

Initial value: $p_1^{BW}(0) = 2$;

Weapon state variable (Ave. # of missiles) per platform: $w_1^{BW}(k)$;

Initial value: $w_1^{BW}(0) = 4$.

Blue Bombers (BB): Ten F4 bomber planes each equipped with 4 MK2s (guided bombs).

Initial Location: $x_1^{BB}(0) = 8$, $y_1^{BB}(0) = 6$;

Platform state variable (F4 bombers): $p_1^{BB}(k)$

Initial value: $p_1^{BB}(0) = 10$;

Weapons state variable (Ave. # of bombs per platform): $w_1^{BB}(k)$

Initial value: $w_1^{BB}(0) = 4$.

8. Concluding Remarks

A nonlinear dynamic model for military operations as a basis for a simulation test bed has been developed. The model is an example of an extended enterprise. The model can be used to investigate different multi-agent control strategies in the presence of a hostile competitor.

9. Acknowledgement

This effort was sponsored by the Defense Advanced Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL), Air Force Materiel Command, USAF, under agreement number F30602-99-2-0549. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the DARPA, the AFRL, or the U.S. Government. The authors would like to thank Dr. Paul Girard of SAIC and Prof. Hiro Mukai of Washington University, St. Louis for fruitful discussions and suggestions.

10. References

[1] Bracken, J. (ed), "Warfare Modeling", MORS, 1995.

[2] Taylor, J.G., "Force-on-Force Attrition Modeling," MORS, 1980.

[3] Dupuy, T. N., "Attrition: Forecasting Battle Casualties and Equipment Losses in Modern War," Hero Books, Fairfax: VA, 1990.

[4] Kott, A., Ground, L., and Langston, J., "Estimation of Battlefield Attrition in a Course of Action Analysis Decision Support System," presented at the *MORS Workshop on land and Expeditionary Warfare*, June 1999.

[5] J. B. Cruz, Jr., and M. A. Simaan, "Multi-Agent Control Strategies with Incentives," *Proceedings of the DARPA/JFACC Symposium on Advances in Enterprise Control*, San Diego, CA, Nov. 15-18, 1999.

Modeling and Control of Competitive Stochastic Processes

Jan Jelinek, and Datta Godbole

Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418
Contact: jan.jelinek@honeywell.com, phone (612) 951-7701

Abstract

Competitors in the marketplace or combatants on the battlefield face very similar challenges: Their resources, be they money or weapons, are gradually attrited in the mutual effort to push each other out of the field and dominate it. Also, even if the participants are deterministic in their decision-making, executing their decisions has random aspects, when the same, generally successful actions occasionally fail for no obvious reasons. The application of system and control theories to improve the planning as well as the plan execution of such processes requires models, which allow planners and managers to reliably predict the expected outcomes of various alternatives over a long horizon into the future. In this article, exact probabilistic models for several classes of battle scenarios are developed from the first principles, which accurately characterize the battle dynamics for arbitrarily long horizons. Then it is shown how the models are used for model predictive control of the battle dynamics

1. Introduction

Combat is an inherently random process. Viewed as concurrent execution of many combats between individual opponents, battle retains this random aspect, which limits what one can realistically expect from battle models. Every battle is a particular realization of the random processes involved, and if the combatants had a chance to fight it over and over under the same rules of engagement, the outcome would vary from one run to another. For example, the numerical superiority of the Blue side is likely to make Blue a winner on average, but cannot save him from occasional losses. Simply put, luck has its role in military affairs. Very much the same story could be said about contract bidding.

When modeling a battle, one can set up a Monte Carlo model, every run of which would generate one realization of the battle, very much like rolling a die generates one out of the six numbers on its faces. Generally, such models are easy to build, because they typically do not involve high levels of abstraction as their structure simply mirrors physical assets along with their geographical layout and interaction links. The randomness of combat is emulated by random generators associated with the interacting assets, whose presence makes the model's states random variables.

When testing various command and control strategies, such Monte Carlo models are indispensable. However, they cannot be used for developing the strategies, nor for amending them on-line when the real time battlefield damage assessment data start coming in. Because strategies are always developed before any actual action takes place, models used for their design cannot work with random variables, but only with their distributions or statistics like expectations, variances and so on, which themselves are not random. Without such predictive models, neither planners nor controllers can be built.

The approach presented in the paper describes the degradation of participants' assets over time as a result of combat activities. The asset degradation may be discrete, with dead or alive being the extreme case of discretization, or continuous. At any time during a battle, each asset is in a particular state of degradation. The set of all possible states of each asset is considered finite. For example, a fighter squadron's state at a given time is the number of aircraft surviving at that time. Due to random effects present in combat, we are unable to predict with certainty the particular states through which the assets will be passing in the course of the battle, but we can derive the probability distributions of the assets' states and their evolution in time as the battle progresses. In [1], the distributions for several classes of battle scenarios are derived from the first principles. It turns out that their time evolution is a Markov process that can be described by difference equations, which are linear in states but nonlinear in the control inputs that the commanders have at their disposal to influence the outcome of fighting, like the deployment of reserves, the use of decoys, the availability and quality of real time damage assessment and so on. One of the immediate uses of the models is their ability to provide quantitative answers about the impact of such inputs on the probability of winning and expected costs of it.

Using these models, we formulate control-theoretic optimization problems that arise in the design of multivariable predictive controllers for competitive stochastic processes.

2. Problem Characteristics

An example problem statement falling into the class of problems solvable within the presented framework may read as follows:

At 0600 the Blue side commander is given the order to completely destroy Red's SAM assets made up of **nLR** *real sites and* **nDR** *decoys by 2400 tomorrow. Because the objective is needed to clear the way for an already planned subsequent offensive, the higher command requests the order be executed with a very high degree of certainty, say less than 1 in 20 chances that it will not be met in full. The Red's SAMs are known to have the lethality* **pLR** *against the attacking aircraft that B is intending to use. They also have a good radar tracking capability to know the accurate numbers and positions of attackers in real time.*

Our solution is addressing in a way, which is optimal in the sense precisely described below, the following questions:

- How does the Blue commander determine how many attack, **nLB**, and decoy, **nDB**, aircraft he needs in his strike package, if his kill rate on the R's SAM's is known to be **pLB**?
- How many missions (sorties), **nRounds**, he should divide his objective into, one, two, or ten?
- If he decides to fly more missions, how he would define their individual objectives, against which he could measure the task's progress once it gets underway? Without them, he would not be able to identify looming problems until it may be too late for any correction.
- If he decides to fly more missions, how to optimally assemble the strike packages for each one? On one side, gradual enemy attrition will lower the threat, but he will have his losses as well. How big? What is the total number of aircraft he should ask to be allocated for the task?
- If, for whatever reason, the task execution does not proceed as planned, what corrective action to take?

Known distance to target allows us to estimate the time needed for a single mission. Let us say that executing one mission takes 6 hours total, i.e., including fly time, refueling and rearming, crew rest, etc. It implies that the task objective must be achieved in **nRounds** = 7 missions or less, if the strike packages can fly round the clock. But that is about the only straightforward part of the solution. Next steps require more sophistication and are outlined below.

3. Modeling Battle Dynamics

No planning or control is possible unless we can predict the expected outcomes of our decisions into the future. Obviously, the longer is the task horizon, the farther out our predictor must reliably go. A predictor, which is unable to forecast the effects of up to 7 consecutive missions cannot be used to solve the given problem in the guaranteed endpoint formulation as requested. One could devise a number of other concepts to circumvent this defect, all of which, in effect, attempt to replace the endpoint control with some moving intermediate point alternative feasible with the available model. One choice, for example, is to strive for certain attrition rates, even if they are not exactly what the commander is really interested in. He knows that attrition is not the same as victory and thus tends to avoid the term so popular with military theorists. Indeed, it is easy to confirm this real life experience by generating Monte Carlo simulations of battles, in which one side maintains its numerical superiority and yet looses large numbers of battles. The underlying cause of this strange phenomenon is the complex interplay of numbers, lethality and the random nature of weapon effects. Averages can be very misleading unless we know the likely spread of actual values about them. The three plots in Figure 1 may be hard to believe to be three different runs of a Monte Carlo simulation of the same battle scenario. In the first plot, B wins hands down in 2 rounds. The outcome of the neck-to-neck fight in the second plot seems to be more a matter of sheer luck than military skill. In the third plot B was really down on his luck, because he was routed by R essentially in the first round.

This example brings up another issue. If the outcomes of applying the same battle strategy to the same enemy can be so vastly different, which one is the right one to choose for making predictions? We can run, say, 1000 battles and obtain the average force strengths or, alternatively, attritions. The strengths shown in Figure 2 show that B will generally maintain his numerical superiority over R throughout the battle. A look at simulation statistics would reveal that this advantage sufficed to help him win about 2 out of 3 battles on average.

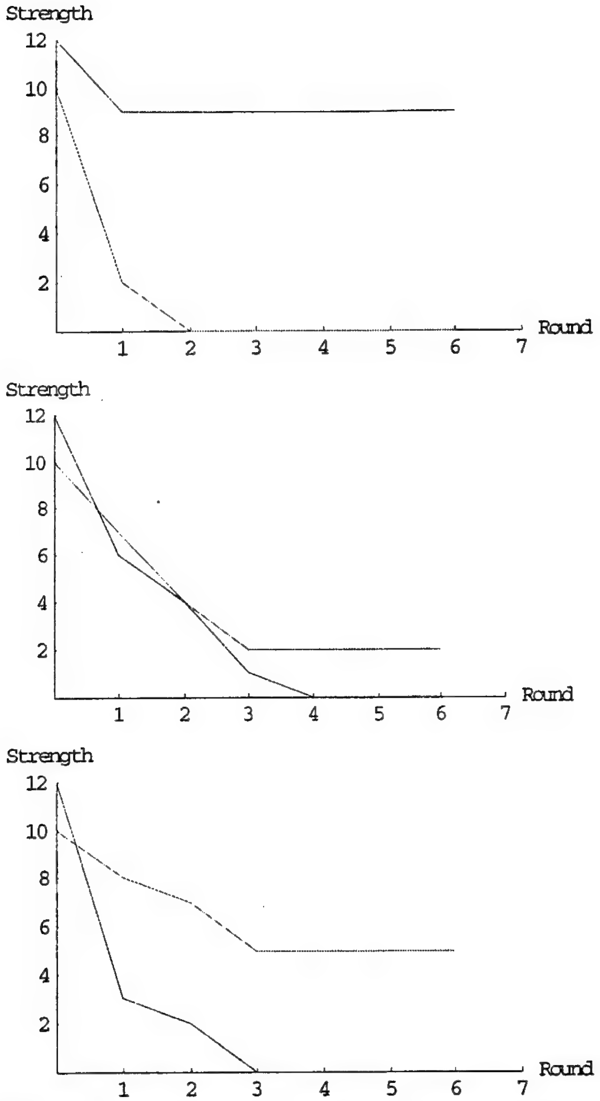


Figure 1: Three Monte Carlo simulation runs of the same battle scenario can yield vastly different outcomes. Due to random weapon effects, luck definitely has its role in any combat. Initially B has 12 aircraft and R has 10 SAMs.

Obviously, if we further increase B's superiority, his odds of winning will improve. At the same time, it is intuitively clear that regardless of his superiority, there will always be a chance, however slight, of loosing due to bad luck. This means that, strictly speaking, we can never provide the 100% guarantee that even the best battle plan and its execution will lead to victory. Moreover, both experience and probability theory tell us that the cost of reducing uncertainty escalates as we approach 0. The number of aircraft needed to destroy a target with the 90% certainty may not be much greater than that for 80%, but going from 99% to 99.9% can be extremely expensive. The lesson to learn from this observation simply is that for stochastic systems (plants) like battles the task objective cannot be meaningfully stated without a desired certainty qualification, because there is no absolute certainty in combat. And this is where we run into difficulties with the Monte Carlo simulations due to the meager amount of information that we can glean from them.

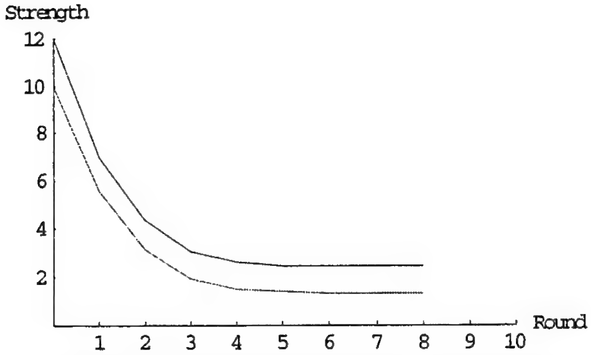


Figure 2: Average strengths computed from 1000 Monte Carlo simulation runs.

3.1. Modeling Battle Dynamics: Predictive Models

Monte Carlo simulations are very easy to set up and can be easily verified for correctness through analysis, because they generally use very little abstraction. One does not have to study them for months to find out how the battlefield reality is mapped into them. Unfortunately, although the information we need for solving the battle planning or control problem in its endpoint formulation could be extracted from the Monte Carlo models in principle, this approach is impractical. It may easily take millions of battle runs even for small numbers of forces involved to obtain reliable enough data allowing us to manage a battle to victory with the quantifiable certainty.

In our earlier reports [1],[2] we have developed predictive models of battle dynamics that are designed for solving this kind of problems. Unlike the Lanchester [4], [5], and other models found in literature, our models are exact in the sense that their predictions of battle state probability distributions exactly agree with experimental data furnished by Monte Carlo simulations not just for a couple of missions ahead, but for arbitrarily long prediction horizons (battle games). We can thus directly use them for the genuine endpoint battle planning formulation as outlined above. The model accepts as its inputs the parameters **{nLB,nDB,pLB}**,**{nLR,nDR,pLR}**. Note that in our model the weapons (aircraft, SAM etc) of B are the targets of R and vice versa. In this model, **nLB** is the number of live weapons of B, **nDB** is the total number of decoys and dead weapons of B, whereas **pLB** is the lethality of B's weapons against R. When the model is exercised, it generates a sequence of probability distributions of possible battle states after repeated missions (strikes, rounds). In the sequence illustrated in

Figure 3, the plot densities are proportional to the probabilities. The top left picture is the initial state of a battle, when both sides know with probability 1 their initial numbers **{nLB = 4, nLR = 11}**. The outcome of the first mission shown in the next picture is not that unequivocal anymore. The most likely number of survivors will be **{nLB = 2 or 3, nLR = 7}**, but other outcomes still rather close to those numbers are possible. As the battle progresses (read Figure 3 row-wise), the cluster spreads more and more until it eventually splits into two, i.e., the distribution becomes bimodal. The last picture in the bottom right corner says that B will win most battles with 1 or 2 likely survivors, but still there will be a significant share of battles won by R. Whenever that happens, R is most likely to end up with 4 or 5 survivors.

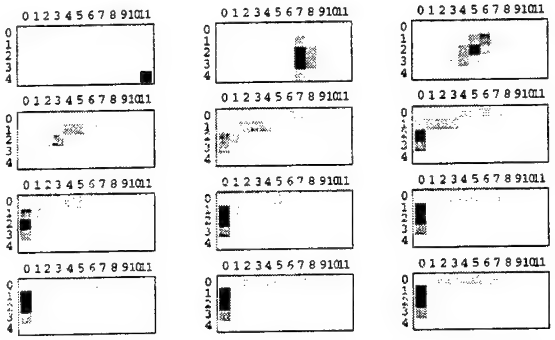


Figure 3: State probability distribution of a battle after 11 rounds (strikes, missions).

The model does not make any assumptions about how its inputs **{nLB,nDB,pLB}**,**{nLR,nDR,pLR}** were obtained, whether they are a product of some sophisticated strategy contrived by a smart enemy commander or just numbers generated at random. To produce the above sequence, we have arbitrarily chosen a very simple rule of engagement for both opponents: In each mission always deploy all survivors from the previous mission, and nothing more or less. Of course, we could have used more complicated rules, which would allow for bringing in reserves, changes in the battle strategy or time-varying weapons lethality due to, for example, day and night mission times, and so on. It is important to keep in mind that our predictive model deals with the statistical consequences of combat carried out under given force specifications, and does not question the specifications themselves. Also the notion of "weapon" is rather abstract: It can be a fighter, bomber, SAM, tank, soldier, etc., so the model is equally applicable to a broad range of military operations as long as they proceed in discrete steps like missions, strikes, sorties, salvos, rounds, etc. When we talk about air strikes here, it is only to facilitate understanding by setting our explanation into a military framework rather than the sole intended model application.

There is one more parameter in the model not yet mentioned, namely the amount and type of real time damage assessment information that the commanders receive. As demonstrated in [1], this parameter has a very strong impact on the battle outcome. Our current models allow the user to choose from the following alternatives:

- Commanders cannot distinguish live targets from dead ones (either killed or decoys).
- Commanders can distinguish live targets from dead ones immediately after each mission.
- Commanders can distinguish live targets from dead ones, but only with time delay amounting to given numbers of missions, which may be different for R and B [2].

Any combination of the above scenarios is also possible, e.g., B gets the feedback information, whereas R does not. The model predicts probability of destroying an asset based on target selection, coordinated weapon use and weapon lethality. In this report we assume that the weapons coordinate their actions, but the target selection process fundamentally depends on battle feedback. Therefore the main parameters of our model are the number of weapons & decoys, weapon lethality and quality of feedback information.

The predictive model used inside the MPC battle controller is described in detail in [1]. It is a Markov process and thus linear with respect to the state distributions:

$$S(k+1) = T \cdot S(k) \qquad (1)$$

S(k) denotes the $(nLB + nDB + 1) * (nLR + nDR + 1)$ state matrix in the k-th round (mission). Formally, the transition matrix has the dimensions $((nLB + nDB + 1) * (nLR + nDR + 1))^2$, and thus its size quickly grows with the combatants' asset size. However, the matrix is very sparse and if the sparsity is properly taken advantage of, the implementation of (1) can still lead to very efficient algorithms, both time- and memory-wise, even for battles involving sizeable assets. Such algorithms were also proposed in [1]. Recently, we have implemented them in Fortan and C, and achieved run times on the order of 1 ms for large problems. Small problems like those used in this report for illustration execute in a time, which is hard to objectively measure due to comparable operating system overheads.

The essence of the model is, of course, in the way the user can set up its transition matrix *T*. The matrix depends on problem specifications

$$T\ \mathbf{▌LB, ▌DB, pLB, nLR, nDR, pDR}\ | \qquad (2)$$

and the theory in [1] derives an exact functional form of this complicated nonlinear relationship. The fact that our model allows to enter its specifications in a form directly related to militarily meaningful quantities instead of some meaningless mathematical coefficients appearing somewehere in differential equations as is the case with the Lanchester model not only adds transparency to it, but allows us to treat the explicit model parameters as control inputs and vary them on-line during the optimization process inside the MPC controller.

Important practical questions are how well such models conform with the reality and how difficult are they to set up. We do not have ready answers to these questions, but in [1] we have suggested a way for predictive model verification that at least appears conceptually plausible. In that concept, a Monte Carlo model serves as an interface between the real battlefield and the abstract worlds of predictions and statistics. We believe that much of the model setup can actually be done automatically from situation and weapons data stored in real time military databases.

3.2. The Lanchester Models

For almost a century, military theorists have been using the Lanchester model to explain the attrition rates observed in actual battles. For battles, in which each side deploys only one kind of resource, its decrease over time is described by the equations

$$\frac{dx}{dt} = -a\, x^{d} \qquad \frac{dy}{dt} = -b\, x^{g} \qquad (3)$$

where x and y are the state variables describing side X and side Y resources, ***a*** and ***b*** are the Lanchester attrition coefficients defining the rate at which Y resources destroy X resources and vice versa, respectively. *d* and *g* are nonnegative exponents, often fractional. Both the state variables and attrition coefficients are assumed to have only nonnegative components.

If each side deploys a number of different resources, the above equations can easily be generalized [4,5].

Many authors have attempted to fit the Lanchester model to real battle data with varying success. The critique of such endeavors is well summed up in [4,5]. It is argued in [4] that the Lanchester model is inherently flawed to capture the reality of combat, and that if there is to be a better model, then the stochastic nature of combat will have to be built into its conceptual fundaments. Often used special cases are *the square law Lanchester model*

$$\frac{dx}{dt} = -a\, y \qquad \frac{dy}{dt} = -b\, x \qquad (4)$$

and *the linear law Lanchester model*

$$\frac{dx}{dt} = -a\, x \qquad \frac{dy}{dt} = -b\, x \qquad (5)$$

Being deterministic, the Lanchester model obviously cannot capture the *actual* attrition rates in any particular battle due to their random nature. At best, we can hope that it can describe the evolution of the *expected* attrition rates. As it turns out even this is too much to expect. Our modeling work [1] shows that even for very simple battles neither the square nor linear law Lanchester model structure is capable of accurately capturing the expected rates, and the models can be considered their approximation at best. Another difficulty with using Lanchester models is the fact that the model parameters do not have domain specific meaning and hence need to be tuned based on the observations of the battle.

4. Model Predictive Control (MPC) of Battle Dynamics

Given the commander's specification as in Section 2, the job of the Model Predictive Control system is to plan and execute the battle as described in Section 2.

4.1. Optimal Deployment of Resources

The model itself does not tell the commander how to prosecute a task in the optimal way, but it is the necessary prerequisite for getting answers. Here is a sampler of possible problems that we can solve:

Optimal Initial Deployment

What is the optimal number of aircraft **nLB** that B needs to deploy in the first mission, if he knows:

1. the enemy specifications **{nLR,nDR,pLR}**,
2. his specifications **{nDB,pLB}**,

3. the rules of engagement direct him to always deploy all surviviving aircraft in subsequent missions with no reserves to be added, and
4. the task is to be accomplished in **nRounds** or less with the desired probability of success **desProb** or higher?

A slightly more complicated version may ask for the optimal numbers of both weapons and decoys **nLB, pDB**.

Figure 4 provides the optimal number of aircraft needed to destroy the enemy given the battle specifications in the number of rounds varying from 1 to 10 with the probability at least **desProb = 0.9**. We clearly see the rapidly growing cost of doing things really fast. We will get back to this figure later in connection with model predictive battle control when we will argue that fulfilling tasks ahead of the optimal plan can be equally detrimental as slipping behind. If the higher command needs a task be done faster, they should say so and the task planner will put together a corresponding accelerated optimal plan. But generally, voluntarism is undesirable, because time saved comes at a cost, too, and unless the superiors have no use for it, acceleration is just waste.

```
[illegible]R, nDR, [illegible] = [illegible], 0.5 ;
[illegible]B, pLB = [illegible] 0.4
```

Program 1. Battle specification used in the examples throughout this report. The symbols **nL** and **nD** represent the number of live and dead (= killed) weapons or decoys, **pL** is the weapon lethality against the expected opponent's weapons

Note that to win in just one round requires 9 to 1 numerical superiority. This may seem a bit too high to the military experts, who thus might question the realism of our predictive model. But keep in mind that it is, in fact, our definition of the battle victory that is somewhat unrealistic, namely the requirement of total destruction of enemy assets with a very high probability. In the military reality, the loosing side disintegrates, if not officially quits, much earlier, when its assessment of winning chances drops below a certain threshold. Thus a more realistic problem statement might be to control the battle toward achieving the probability of win, say, 80%, instead of the total force annihilation. Although we have not done the calculations, we expect this step would bring the numbers down considerably and put them in a better agreement with the military experience.

Optimal Intermediate Deployment

What are the optimal numbers of aircraft **nLB(k)** that B needs to deploy in each (k-th) mission from the beginning to end, if he knows:

1. the enemy specifications {**nLR(k),nDR(k),pLR(k)**} for each mission,
2. his specifications {**nDB(k),pLB(k)**}, and
3. the task is to be accomplished in **nRounds** or less with the desired probability of success **desProb** or higher?

Again, a more complicated version may ask for the optimal numbers of both weapons and decoys **nLB, nDB**.

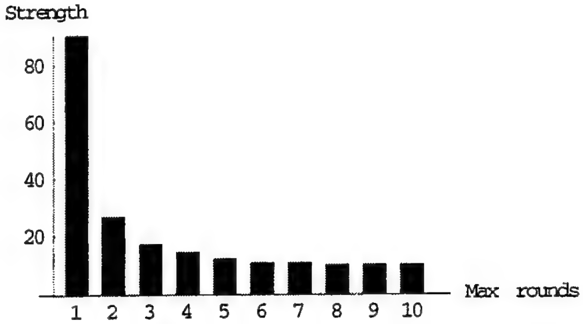


Figure 4: Numbers of aircraft needed to accomplish a given task for varying number of missions.

As the reader may have noticed in the optimization problems above, there was no mention of own losses. Achieving the objective was the only aspect that mattered and if we lost x aircraft along the way, so be it. There certainly are such urgent tasks in wars, but more typically, there will be concerns about losses. Here we are intentionally avoiding the notion of cost, which all classical, deterministically formulated optimization problems introduce as a necessary technicality, but which has a little meaning in war. One may say with some exaggeration that dollars do not make sense in war, and resources and targets do not have their military value price tags painted on them. The probabilistic formulation again offers a more realistic look at the problem.

Once we bring own losses into consideration, the optimization problems take on a different twist. One can easily imagine two extreme alternative solutions to the SAM destruction task cited earlier, and then many others in between. One simply assumes that B takes the enemy's weapon lethality **pLR** against his attacking aircraft as a fact of life which he can do nothing about, hence he is going to bite the bullet and use the optimal number calculated by one of the above algorithms. The other is based on assumption that B can actually proactively manipulate R's weapon lethality. For example, adding radar jamming support aircraft to his strike package will reduce SAM's lethality and his losses will drop. If adding one such airplane cuts **pLR** by 0.1, how many of them does B need to limit his attack aircraft losses, say, to under 5 pieces and still accomplish the original objective?

Optimal Strike Package Composition

What are:

1. the optimal numbers of attack aircraft **nLB(k)** that B needs to deploy and
2. the minimum reduction of the R's weapon lethality **pLR(k)** that B needs to achieve in each (k-th) mission from the beginning to end,

if he knows:

1. the enemy specifications {**nLR(k),nDR(k),pLR(k)**} for each mission,
2. his specifications {**nDB(k),pLB(k)**},
3. the task is to be accomplished in **nRounds** or less,
4. own losses cannot exceed **lossB**

and all that must be met with the desired probability **desProb** or higher?

Here we assume that based on experience, B can translate the required drop in R's lethality into a corresponding number of support aircraft.

4.2. Closed Loop MPC

No feedback control, also called open loop control by control theorists, computes the optimal deployment of forces only once, before the task execution starts. Once it gets going, it lets its sequence of missions run its own course, without any further intervention. Commanders simply keep sending back into action all surviving aircraft until one side looses all. It was this kind of control, or rather lack of it, that was used to produce the plots in Figure 1 and 2.

Now imagine that B has some reserves that he can bring in if bad luck drives him off the planned course. On the other hand, when luck has him do better than anticipated, then he can put the unneeded aircraft back into the reserve pool to make it available to others rather than finishing his task ahead of the plan. Using one of our predictive model based optimizations listed above, we can now easily build a model predictive controller (MPC) of battles. For the sake of simplicity, we have opted for the optimal initial deployment optimizer to be used at its core, but any one from Section 4.1 could have been used as well (and many others), with the corresponding benefits. As we shall see, despite its simplicity, the *Initial Deployment MPC* performs extremely well and shows considerable robustness with respect to model-plant mismatches.

We shall explain the Initial Deployment MPC concept by contrasting it against the open loop (= no control) solution.

Prior to the first mission, both the no-control and MPC-control strategies do the same calculations, namely compute the optimal number of aircraft to be flown in the first round (mission) using the initial deployment optimizer. Even though the optimizer explicitly returns only one number, namely the number of aircraft to be used in the first mission, it actually computes the solution all the way up to the victorious endpoint assuming that all survivors will always be redeployed in full. If B wants to know the expected losses in each round, both his own and R's, he can obtain them easily by exercising the predictive model using the optimal number **nLB**.

Because both strategies use the same optimization algorithm, they come to exactly the same conclusions. Therefore, the first mission strike package makeups are always identical. After the first mission, however, they will start to differ. The no-control commander will thoughtlessly gather all his surviving resources and order them to fly the second mission. The MPC-controller (or, better, the MPC-advised commander) will first look at the damage assessment intelligence and critically reevaluate his standing. For this purpose he will use the same model as before, but will enter the intelligence updates on the actual enemy strength after the first mission and will also reduce by one the maximum number of missions allowed to fulfill the task objective. This reevaluation will generally produce slight corrections to the actual number of survivors that the no-control commander will use, because our first mission plan could not know what would exactly happen in combat and thus worked only with outcome probabilities. Feedback is thus closed through the ongoing replanning and implemented as corrections to package composition. The corrections are either drawn from or returned to the pool of reserves.

4.3. Experimental Results

In this section we present results of numerical experiments where the Blue MPC controller was driving a Monte Carlo battle simulator fighting the Reds, whose commander was dutifully following his orders to always redeploy all survivors. Vertical bars represent the reserves added to or withdrawn from the survivors. The first mission bars are the initial deployments, and thus must be identical in all battles. There are no other red bars, because R makes no follow-up corrections to the survivors. The thin lines are the number of survivors after each mission, the dots mark the actual number of aircraft deployed. Note that the red dots always lie on the same horizontal level as was crossed by the thin survival plot in the preceding mission. Due to his ability to add or withdraw airplanes, this is not generally true for B, for whom the sum of previous mission survivors and reserve change in the current mission determines the blue dot's vertical position. Figure 5 plots average strength from the statistics collected on one batch of 1000 randomly generated battles fought under the same scenario as before. The most remarkable observations are that:

1. MPC did not loose a single battle, and
2. on average, it kept withdrawing rather than adding reserves.

The latter indicates that in addition to the outstanding performance, one of the rather unexpected benefits of MPC control is a better utilization of resources.

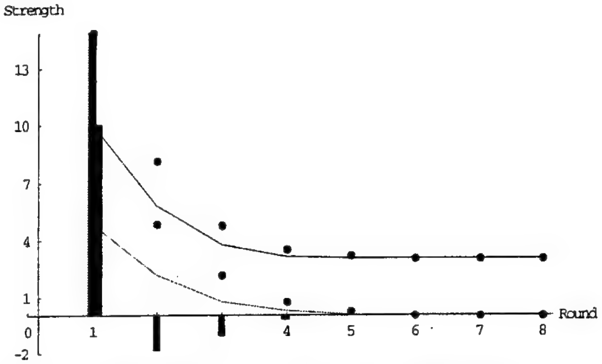


Figure 5: Averages from an ensemble of 1000 randomly generated experimental battles. As before, bars, dots and solid lines represent deployment increments/decrements, actualy deployed and surviving aircraft in each mission, respectively.

As the following Figure 6 illustrates, both the MPC-controlled and no-control battles have losses identical within the statistical margin error.

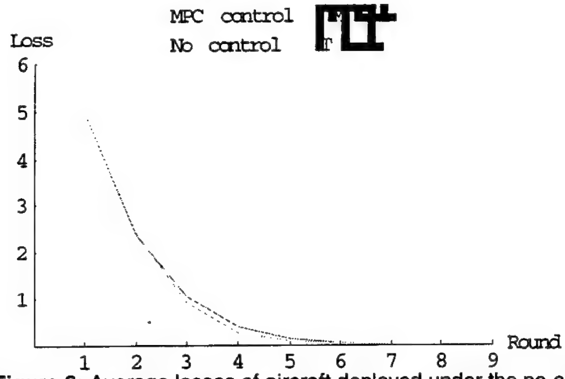


Figure 6: Average losses of aircraft deployed under the no-control (turquois plot) and MPC (magenta plot) strategies.

For comparable losses, however, we get very different return on our resource investments. The average numbers of deployed aircraft per mission without control and with MPC control plotted in the Figure 7 convincingly demonstrate the wasteful use of resources in the no-control case.

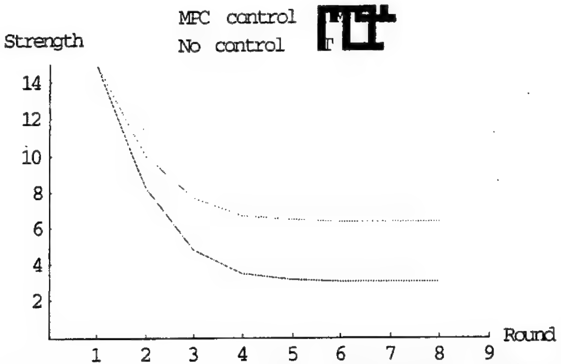


Figure 7: Average numbers of aircraft deployed under the no-control and MPC strategies.

This contrast is further amplified by the fact that while the MPC battle controller did not loose a single battle in the ensemble of 1000 (although it may very rarely happen), with no control the probability of losing is not even close to zero. That is, without control, on average, we can expect either to loose or not win within 10 rounds about 66 battles in every ensemble of 1000.

5. Robustness of the MPC Battle Controller

Robustness refers to the resilience of the model predictive controller performance to mismatches of its internal predictive model with the reality. We have experimentally investigated two kinds of plant-model mismatches:

1. The MPC model under- or overestimates the lethality of R assets,
2. The B's real time damage assessment intelligence is inaccurate and under- or overestimates the size, **nLR**, of R assets.

Results from thousands of Monte Carlo simulations of various degree of mismatch convincingly demonstrate remarkable immunity of the Initial Deployment MPC battle controller to inaccurate battlefield information. However, as with everything else in life, the ignorance does have its price: The MPC controller will keep winning, but B will pay for his victories with either increased losses or higher opportunity costs for wastefully using his aircraft to no extra benefit. The following subsections offer the gist of the results.

5.1. B Underestimates the Lethality of R Assets

Figure 5 through 7 illustrate the performance of the MPC controller, which plays the role of the B commander, providing that its model of R perfectly matches his actual strength and lethality. Below in the Figure 8 results are presented for the case when the actual lethality is **pLR** = 0.5 as before, but B believes that it is only **pLRm** = 0.2.

In this particular ensemble of 1000 battles, B happened to lose 4 of them, which seems to be still acceptable performance degradation. This number can slightly fluctuate for other ensembles due to the random nature of Monte Carlo simulations, but generally will be in the range of a couple of losses. As we can see in Figure 8, B starts with only 10 airplanes in the first mission, but his incorrect lethality estimate forces him to bring in more and more reserves in each of the subsequent missions and, at the end, pay for his error with higher expected losses. It is interesting that they are higher in spite of the total number of deployed aircraft being lower. Also note that the expected number of missions B needs to win goes up.

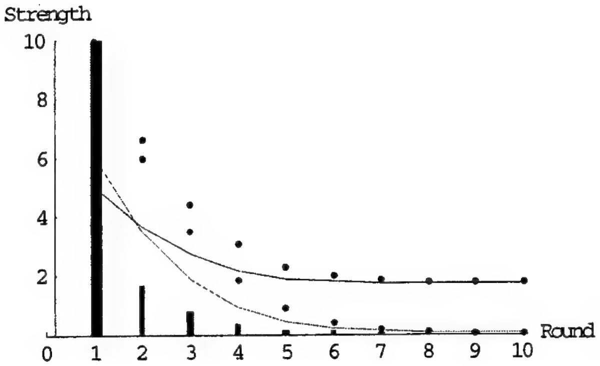


Figure 8: Averages from an ensemble of 1000 randomly generated experimental battles, in which B mistakenly guesses the R weapons' lethality at 0.2, while it actually is 0.5.

We can better appreciate the excellent MPC performance once we realize that the loss of only a couple of battles out of 1000 is nothing compared to the disaster which such a gross underestimation would have led to without control. In that case, the probability of B loosing is 0.747802, i.e., B can expect either to loose or not win within 10 rounds about 748 battles in every ensemble of 1000.

5.2. B Underestimates the Strength of R Assets

Underestimating the enemy's strength has similar effects as underestimating his weapons' lethality. In each subsequent mission, B has to keep adding airplanes from his reserves in his attempt to meet the task objective (see Figure 9). In each mission, his plan undergoes significant revisions, and yet he never quite catches up with the reality. As the statistics below show, if his damage assessment underestimates R's numbers by 50%, then he would end up loosing 51 battles out of 1000. (There was one draw in this particular ensemble). As bad as it looks, it is actually a testament of the excellent controller performance. Without MPC control, B would have lost about 952 battles in every ensemble of 1000 on average. He pays a premium for his victories, though. This shows the importance of intelligence in winning a battle.

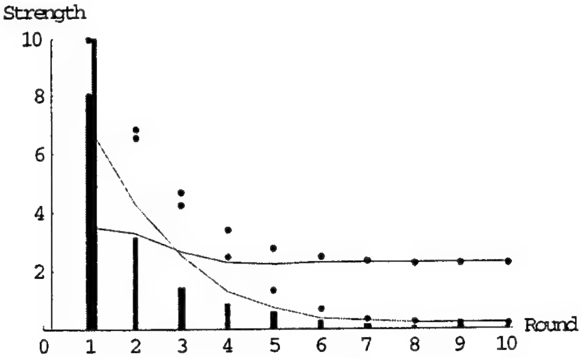


Figure 9: Averages from an ensemble of 1000 randomly generated experimental battles, in which B receives damage assessment intelligence underestimating the R's surviving numbers by 50%.

Similar results [8] are obtained when B overestimates enemy lethality or numbers, except that in this case MPC starts out with excess deployment of resources and ends up withdrawing many of them as the battle progresses. Our experiments show the importance of obtaining correct battle damage assessment and intelligence information. It also shows that the feedback control using the proposed MPC formulation reduces sensitivity to model mismatch.

6. Summary and Future Work

We presented design and experimental testing of the Initial Deployment MPC Battle Controller, which enables the commander to conduct the battle associated with a given task so as to achieve its military objectives with the desired certainty and within the given deadline. As we have demonstrated in thousands of Monte Carlo experiments, the controller hardly looses a battle, even if its information about the enemy is not particularly accurate. Because its victories are always accomplished with the minimum number of resources needed for the given task, its application in concurrent tasks increases the utilization level of military resources by allowing them to be deployed in tasks where they have the biggest impact. We are currently focused on the following extensions.

Optimal Strike Package Composition
The concept is clear and has already been described in Section 4.1. The MPC algorithm has been developed and its implementation along with the experimental results are contained in the report [7].

Model-based Damage Assessment Intelligence Evaluation The availability of the predictive models of battle dynamics opens up a whole new way of dealing with the intelligence data. There is no need anymore to accept the data at their face value, but subject them to ongoing "reality checks" by comparing them with the expectations. For example, if the MPC comes up with its model-based forecast of, say, 2 expected losses out of 10 deployed aircraft in the upcoming mission, then the actual loss of 1 or 2 or 3, or perhaps even 4 planes can be interpreted as a mishap due to random fluctuations and would not necessarily raise questions concerning its model validity. However, if the strike package returns decimated by 8 planes, then one is likely to start wondering if there is something wrong with the data or our model of the enemy's capabilities. How can we find out?

As it happens, this problem is well studied in statistics and goes by the name of statistical hypothesis testing. The catch is that one needs to know the probability distribution of outcomes for the problem to have a solution. If the sample is large like in newspaper polls, then one can safely consider the data constituting a sample to obey the normal distribution regardless of the distribution that actually governs the sampled population. This is definitely not the case, though, in our framework. Each mission contributes only one piece of data to our sample, namely the number of enemy survivors, **nLR**. We cannot wait until we gather hundreds of samples, because the battle will be over in a few rounds and thus they will never come. On the contrary, we need to validate the intelligence data continuously and immediately after the first, second, and so on, mission, while we still have an opportunity to benefit from it. Such small samples require the actual probabilistic distribution of the data generating population, which is exactly what our predictive models can provide. As it happens, the distribution varies with each additional mission, thus rendering the usual textbook statistical hypothesis testing procedures useless.

Resource Apportionment among Concurrent Tasks

So far we have dealt with the solution of a single task. In the real world, there will be many tasks running concurrently or at least making claims on their share of available resources. In our vision, each task will be assigned its MPC controller, which by the way of its operation constantly computes and updates the likely number of remaining missions and expected deployments for each. A straightforward extension is to let the controller also calculate the sensitivity of its strategies to not meeting the expectations on resources. These data will constitute inputs to our resource apportionment optimization, which has been implemented and initial results reported in [6].

Controlling Battles Against Intelligent Adversary

So far we have assumed that the B commander rigidly follows his rules of engagement that B happens to know. There is nothing stochastic in the R's rules. For each possible battle state, they prescribe the R commander how to respond, and he dutifully acts as a stimulus-reaction machine driven by a program, which B knows as well. In this model, combat is the sole source of unpredictability, which takes on the form of randomness.

However, the knowledge of the enemy's playbook is rather unusual in the real world. Even if we know the rules of engagement, they always leave enough room for commander's creativity to fool his adversary. Under such circumstances, we cannot really say with certainty, what R is going to do in response to a particular situation. Instead, we can ask what he is potentially up to given his resources and then assume that he will try to use them in a way that harms us the most. This is the game-theoretic formulation of the enemy's behavior, which we will investigate. In technical terms, it leads to the same optimization problem formulations as listed in Subsection 4.1, but with modified criteria.

Acknowledgements: This document is based up on work supported by the DARPA JFACC program through SPAWAR Systems under contract no. N66001-99-C-8501. Any opinions, findings, conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of DARPA and SPAWAR Systems.

Bibliography

1. Jelinek, J. " Predictive Models of Battle Dynamics", JFACC project report, Honeywell HTC, February 2000
2. Jelinek, J. " Predictive Models of Battle Dynamics: Part II - Delayed Feedback", JFACC project report, Honeywell HTC, April 2000
3. Hillestad, R. J. and Juncosa, M. L.: " Cutting Some Trees to See the Forest: On Aggregation and Disaggregation in Combat Models", in Bracken, J. et al, eds.: Warfare Modeling, John Wiley & Sons 1995
4. Ancker, C. J., Jr.: " A Proposed Foundation for a Theory of Combat", in Bracken, J. et al, eds.: Warfare Modeling, John Wiley & Sons 1995
5. Hartley III, D. S.: "A Mathematical Model of Attrition Data", in Bracken, J. et al, eds.: Warfare Modeling, John Wiley & Sons 1995
6. Tierno, J. "Distributed Multi-Battle Model Predictive Control", JFACC Project Report, Honeywell HTC, June 2000
7. Jelinek, J. "Winning Battles Without Losses", JFACC Project Report, Honeywell HTC, June 2000.
8. Jelinek, J. and Godbole, D.: "Model Predictive Control of Battle Dynamics", JFACC Project Report, Honeywell HTC, May 2000.

Section 4

Variable Granularity Models: Abstractions & Decompostions

Hierarchical Consistency of Supervisory Command and Control of Aircraft Operations

Wang Xi, Asok Ray†, Hui Zhang
Mechanical Engineering Department
The Pennsylvania State University
University Park, PA 16802
† email: axr2@psu.edu

Shashi Phoha
Applied Research Laboratory
The Pennsylvania State University
University Park, PA 16802
email: sxp26@psu.edu

Abstract

A major goal of Command and Control (C^2) in air-battle management is to achieve the mission of a squadron consisting of several aircraft. A hierarchically structured C^2 system of aircraft operations has been synthesized in the discrete-event setting based on finite state automaton (FSA) models. The lower tier of this supervisory control system consists of several logically parallel units, each representing a discrete event model of an autonomous aircraft and its own local controller. An information channel filters the outputs of the (controlled) behavior from each aircraft based on the fact that the upper tier does not need to exercise control on each action at the lower tier. Therefore, the atomic events at the upper tier are constructed as compositions of lower-tier events in the sense that a higher-level language instruction is a composite of multiple machine-level instructions that are executed on parallel finite state machines. The composite behavior of these parallel machines constitutes a virtual plant model for synthesis of the upper tier controller. This paper presents a construction mechanism for formulating control specifications for hierarchically structured controllers and addresses some of the associated theoretical issues in the context of a multi-aircraft air campaign including control specifications for multi-aircraft operations.

Keywords: Command and Control; Supervisory Control; Discrete Event System

1. Introduction

Hierarchical decomposition is known to reduce the order of complexity for synthesis of decision and control problems [3]. The hierarchical control of Discrete Event Systems (DES) is built upon the concept of hierarchical consistency in the framework of Ramadge and Wonham [2] that is referred to as the *RW* framework in the sequel. This concept presents the strict-output-control-consistency conditions that guarantee high-level abstraction can be obtained and a (supremal) low-level implementation of the high-level control exists. The process of abstraction is represented by vocalization of strings. However, the description of the high-level abstraction is highly involved and the physical meaning in such a refinement process is intractable in certain cases. In addition, the blocking issue in the low-level implementations was not considered by Zhong and Wonham[7]. The motivating work on hierarchical structure of DES is presented by Wong and Wonham [5,6]. The concept of control structure generalizes the *RW* framework in the sense that the hierarchical control problem can be solved by the same concept of controllable sub-languages and is guaranteed to be consistent. Wong's work was the first that gave the interconnection between observability and the non-blocking property. A critical point of a successful hierarchical control design is the proper definition and exploitation of the observer which provides the key conditions for architectural decomposition subject to the requirement of non-blocking. However, due to the algebraic nature of Wong's work, it is still hard to work within real implementation. Wong and Wonham [5,6] provide fairly applicable conditions on abstraction of the low-level models in order to enforce safety and liveness specifications at all levels of the hierarchy. They further introduced the concepts of consistent abstraction and reliable abstraction for the high-level virtual plant construction and pointed out that only reliable abstraction can guarantee the hierarchical consistency when there is interaction among component systems at that level of the hierarchy.

Parallel to the event-based treatment of hierarchical control is the state-based approach of Caines and Wei [1], which presents a bottom-up abstraction technique using state aggregation modeled by a partition of the state space. To obtain a high-level transition structure, a so-called dynamical consistency condition on the partition is formulated. It can be shown that the event-based approach is equivalent to the state based approach in the sense of hierarchical consistency condition.

The hierarchical structure of the DES supervisory control, developed in this paper, is formalized in the automaton-based *RW* framework by following the concept of bottom-up model construction and top-down control of hierarchies [6]. Specifically, the dynamics of aircraft operations are modeled in Finite State Automata (FSA) representation, and a maximal permissive supervisor is synthesized based on the desired system behavior, i.e., a given set of specifications.

This paper is organized in four sections including the present one, and an appendix. Section 2 presents DES modeling of aircraft operations and describes desired system behavior. Mathematical preliminaries for synthesis of DES control systems are presented in Appendix A. Section 3 introduces the methodology of controller synthesis. This paper is summarized and concluded in Section 4 with recommendations for future work.

2. System modeling and control objectives

The Command and Control (C^2) system involves different types of platforms and weapon systems for air operations. In this paper, we present a simplified model of a combat aircraft, popularly known as wild weasel that is capable of both aerial battle and attacking ground targets. The DES control system under consideration must be controllable and non-blocking to ensure that the control system will have the capability to manipulate aircraft operations to fulfill the mission unless destroyed or forced to abort the mission.

Our approach is to synthesize a hierarchical control as opposed to a centralized controller. For example, if a (non-hierarchical) control structure is obtained by synchronizing models of all plants (i.e., individual aircraft operations) to design a centralized controller, then the synthesis process is likely to suffer from an exponential state-space explosion. In the present work, we have taken the advantage of both vertical (i.e., hierarchical) and horizontal (i.e., modular) mission decomposition. The approach embodies several low-level real world models, G_{LO}'s, controlled by the corresponding localized controllers C_{LO}'s and a high-level virtual model G_{HI} controlled by a global supervisor C_{HI} as seen in Figure 1. These low-level controllers achieve individual local goals while the *global* goal is assigned to the high-level controller. In reality, the high-level virtual model G_{HI} does not execute the control actions of C_{HI}; they are passed down to respective C_{LO}'s that are commanded by C_{HI}. In essence, C_{LO}'s follow the commands issued by C_{HI}.. In the context of hierarchical control, an upper tier event that is disabled by C_{HI} can be implemented by one or more C_{LO}'s by disabling one or more corresponding lower tier events. Hierarchical control systems synthesis must ensure consistency to achieve the global mission in air operations such as a squadron of several aircraft employing hierarchical mission decompositions. Inconsistency may lead to irrational behavior of an individual aircraft. [Note: The decision problem of this two-tier control system is defined to be consistent if the high-level controller through coordination among the low-level controllers achieves the overall goal.]

The feature selector in Figure 1 is realized by a mapping $\theta : \Sigma_{lo} \rightarrow \Sigma_{hi}$, which reduces the information flow between the upper-level controller and the low-level

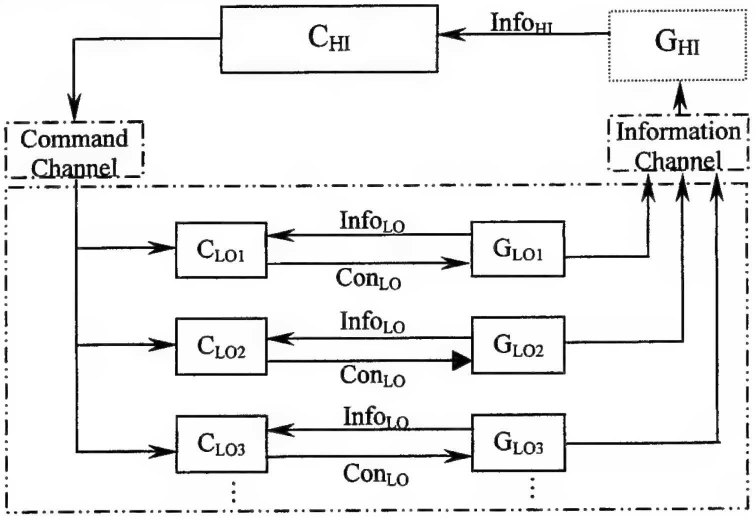
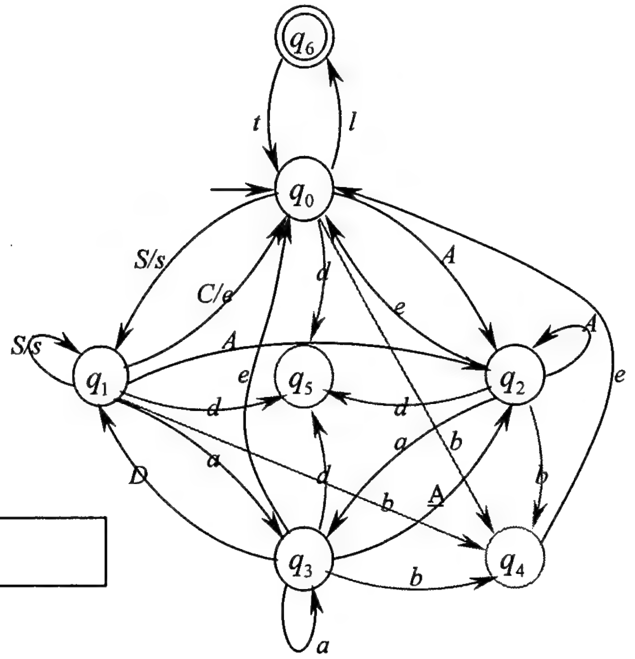


Figure 1. Hierarchical structure of a two-tier supervisory C^2 System

$\lvert\delta\rvert$	In	Out	$\lvert\delta_{q_i}\rvert$
q_0	5	5	10
q_1	3	5	8
q_2	3	5	8
q_3	3	6	9
q_4	4	2	6
q_5	5	0	5
q_6	1	1	2
$\sum_i \lvert\delta_{q_i}\rvert$			24

$$\Sigma_c = \{a, e, l, S, s, t, r, R\}$$



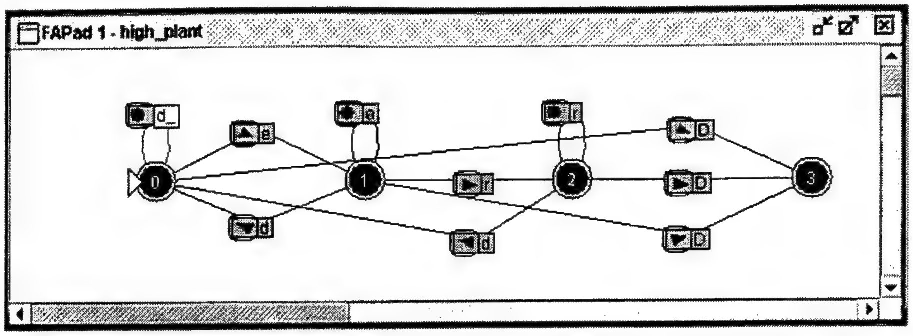
EVENTS	PHYSICAL MEANING	STATES	PHYSICAL MEANING
a	attack the target, *i.e.*, fire	q_0	Idle in air and safe (ready for mission)
A	alarm, in the range of the target	q_1	Searching for target
b	partial damaged	q_2	Alarming that the aircraft is in danger
C	mission completed	q_3	Firing the missile
d	destroy	q_4	Damaged, can fly but cannot fight
D	destroy the target	q_5	Get destroyed completely
e	escape	q_6	Mission completed/abort, back at base
l	all targets destroyed/mission abort		
S/s	search target/friend		
t	take off from the base		

Figure 2. FSA model of a Wild Weasel aircraft

controller in the sense that several low-level events are aggregated into a high-level event. Accordingly, several low-level states have been aggregated into a high-level state. In the implementation if the image of two consecutive low-level events are the same, by feature selector θ, there will be only one upper-level event which goes into the upper-level controller.

A detailed DES model of Wild Weasel aircraft operations, known as the plant model and its controller, has been developed for simulation experiments. For the purpose of illustration, this paper uses a simplified DES model G_{LO} of aircraft operations as shown in Figure 2.

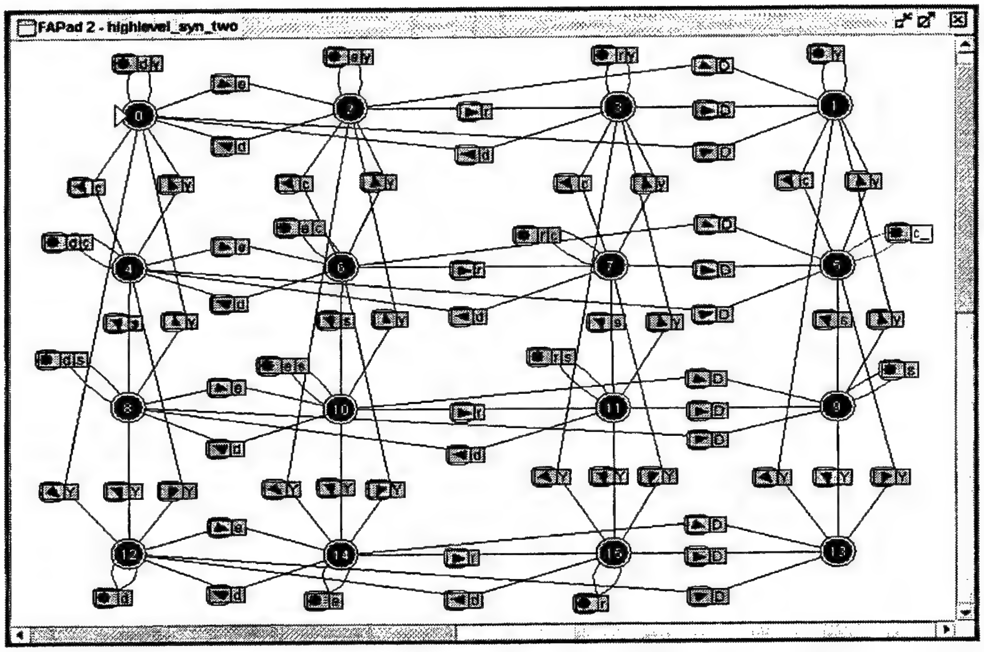
The high-level virtual plant model G_{HI} is obtained as asynchronous composition of two identical models (G_{LO}'s) of the aircraft as shown in Figure 3. The low-level localized controller (C_{LO}) for each aircraft and the high-level supervisor (C_{HI}) for a group of two such identical aircraft are presented in Figures 4 and 5, respectively.



High-level virtual model for each wild weasel after feature selector θ (•)

EVENTS	PHYSICAL MEANING	CONTROLLABLE	FEATURE SELECTOR
e	engage	T	{s, S, A, m}
d	disengage	T	{C, u, v, e}
r	rescue search	T	{s}
D	destroy	F	{d}
States	**Physical Meaning**	**States aggregation of lower level plant**	
q_1'	disengaging	$\{q_0, q_1, q_6\}$	
q_2'	engaging	$\{q_2, q_3, q_4\}$	
q_3'	rescue searching	$\{q_1\}$	
q_4'	destroyed	$\{q_5\}$	

Definitions of events and states for each wild weasel in the high-level virtual plant model



Synchronous composition of two high-level virtual plant models

Figure 3. High-level virtual plant construction

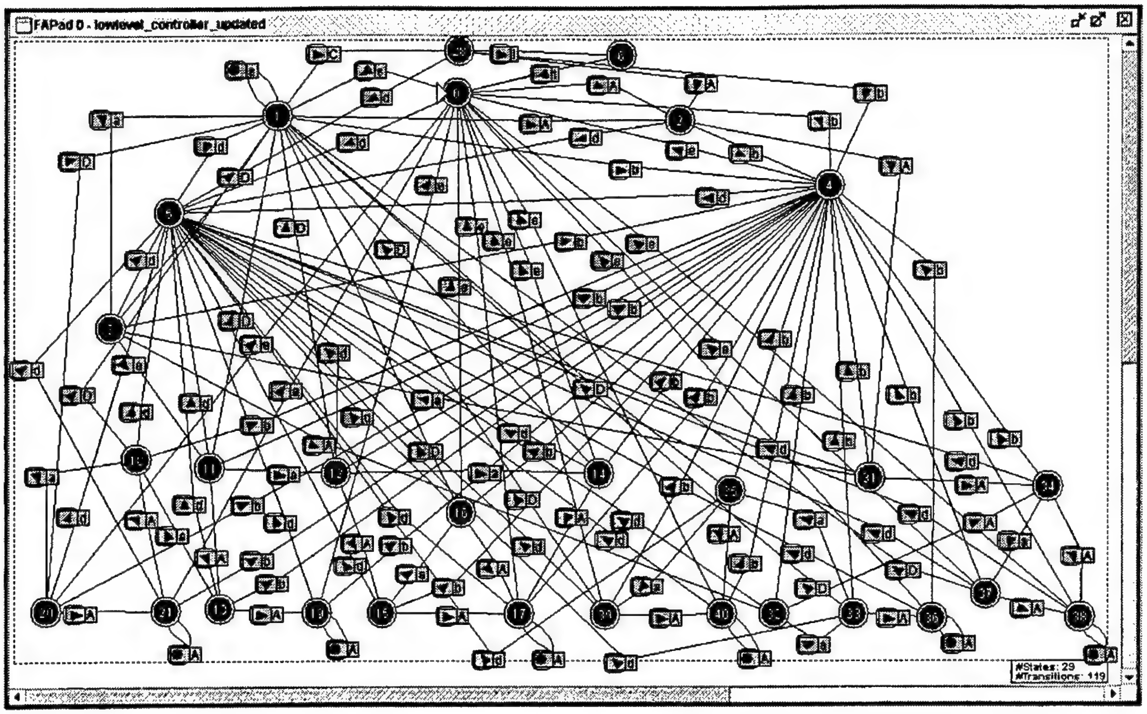


Figure 4. Low-level localized controller for each Wild Weasel aircraft

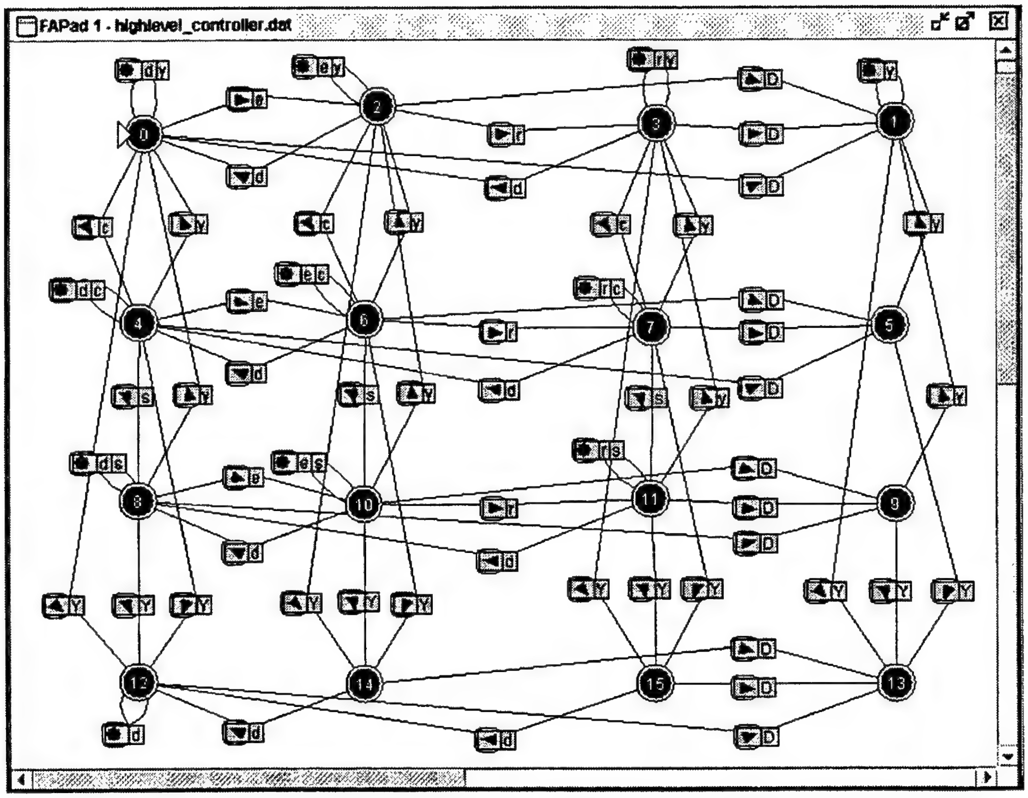


Figure 5. High-level supervisor for a group of two identical Wild Weasel aircraft

3. Design methodology

The proposed hierarchical structure in Figure 1 is constructed bottom-up and is controlled top-down. All the control specifications in the regular language fashion are prefix-closed and the corresponding FSAs are trim, thus blocking is not an issue. In addition, we assume complete observation of the event generation. We proposed the design procedures for the 2-tier hierarchy as follows:

Low Level Control Specifications (for operation of individual aircraft):

Try to fulfill the mission if none of the following situations occurred:

- When at the initial state q_0, start searching the target if the mission is not completed;
- Start attack after two or more consecutive alarm signals;
- Escape if consecutive alarms from the enemy exceed 4 times when in alarming state;
- Escape if consecutive attacks to the target exceed 4 times when in attacking state;
- The (partially) damaged aircraft should abort the attack and escape if capable to do so.

High Level Control Specifications (for operation as a Squadron Leader):

Try to fulfill the mission if none of the following situations occurred;

- If an aircraft is down and if there is a single remaining aircraft, then it may initiate rescue search before escaping;
- If one or more aircraft are partially damaged and if there is a single remaining aircraft, then it may escort the damaged aircraft while escaping.

Design of an Individual Controller (C_{LO}):

- Model the plant as a finite state automata, *i.e.*, $G=(Q,\Sigma,\delta,q_0,Q_m)$;
- Convert the control specifications into a finite state automata, *i.e.*, S = (X, Σ, σ, x_0, X_m), such that $K=L_m(S)$;
- Check if G is accessible, *i.e.*, if all states in G are reachable from the initial state q_0;
- Check if S is trim, *i.e.*, S is both accessible and co-accessible;
- Check if K is prefix-closed, by checking this property, we can be sure if there exists potential blocking issue;
- Test controllability of K with respect to G. That is, $pr(K)\Sigma_u \cap L(G) \subseteq pr(K)$;
- If K is controllable, choose S as the supervisory controller;
- If K is uncontrollable, repeat the procedure to compute a supremal controllable sublanguage.

Hierarchical Controller Design

- Decompose the mission at the low level to obtain local control specifications;
- Construct local plants $G^i{}_{Lo}$, and specifications $S^i{}_{lo}$ as FSAs such that $K^i{}_{lo} = L(S^i{}_{lo})$,, i=1,2,...n;
- Check controllability of specifications $K^i{}_{Lo}$ with respect to the corresponding plant $G^i{}_{Lo}$. If uncontrollable, compute the corresponding supremal controllable sub-language sup $C(K^i{}_{Lo})$;
- Close the loop to obtain the synchronous composition $K^i{}_{Lo} \| G^i{}_{Lo}$ for each low level mission;
- Abstract information on n subsystems as: $G^i_{Hi} = \theta\left(K^i_{Lo} \middle\| G^i_{Lo}\right)$ by designing the feature selector $\theta : \Sigma_{Lo} \to \Sigma_{Hi}$;
- Obtain a synchronous composition of the n subsystems $G^i{}_{Hi}$ as $G^i{}_{Hi}$;
- Test controllability of K_{Hi} with respect to G_{Hi}.

In the feature selector design, it must be guaranteed that no conflicts could exist between supervisors at the low-level and the high-level because the high-level virtual model is obtained in such a way that internally the low-level specification has been incorporated. This step is quite similar to the hierarchical controller design of continuous-varying system. The above procedures can be extended to n-level hierarchical control structure based on careful mission decomposition of the physical problem. So far, we've designed a highly efficient controller synthesis tool in the JAVA language. The detailed development of the package will be described in a forthcoming publication.

For implementation, the entire hierarchical system is interactive between the discrete event controller and the continuously varying process of aircraft dynamics. The importance of the event generator and the action generator has been brought to be a key point. A critical question is: how frequently do we need to generate the 'Alarm' event once the aircraft gets into the range of the target? A possible answer to this question can be obtained by having two parallel infrastructures of command and information. Both of them must run simultaneously since the DES controller does not have the capability of issuing a command with detailed information for execution of the aircraft operations. For example, once the DES controller issues a command 'attack', it is the tactical intelligence/information infrastructure's responsibility to give the detailed coordinates to be fired. Similarly, the tactical intelligence should tell the event generator when the aircraft finds the target after the controller issues a command 'search for the target'. Currently, the low-level controller has been shown to be properly functioning and the high-level controller is being implemented.

4. Summary, conclusions, and recommendations for future work

This paper addresses hierarchically structured Command and Control (C^2) of aircraft combat operations in a discrete-event setting based on finite state automaton (FSA) models. The goal of the C^2 system under consideration is to achieve the mission of a squadron consisting of several aircraft. The lower tier of the two-tier C^2 system consists of several logically parallel units, each representing a discrete event aircraft model G_{LO} and its own local controller C_{LO}, in the setting of supervisory control. An information channel filters the outputs of the (controlled) behavior from the aircraft based on the fact that the upper tier does not need to exercise control on each action at the lower tier. Therefore, the atomic events at the upper tier are constructed as compositions of lower tier events in the sense that a higher-level language instruction is a composite of multiple machine-level instructions. These machine-level instructions are executed on parallel finite state machines G_{LO}. The synchronized composite behavior of these parallel machines constitutes the virtual plant model G_{HI} for synthesis of the upper tier controller C_{HI}. Since the control actions of C_{HI} cannot be executed by G_{HI}, they are passed down to respective C_{LO}'s that are commanded by C_{HI}. In other words, C_{LO}'s carry on the commands issued by C_{HI}.. In the context of supervisory control, an upper tier event that is disabled by C_{HI} can be implemented by C_{LO}'s by disabling one or more corresponding lower tier events.

This paper shows how to synthesize a supervisory controller under complete observation within a hierarchical structure of air operations where the system components are modeled as discrete event systems. The advantages of the proposed C^2 system architecture include:

- Logical partitioning of the control task into different tiers, with the lower tier controlling the detailed behavior of each aircraft and the upper tier fulfilling the mission objectives.
- Reduction of computational complexity in the sense that the upper tier controller is synthesized over the virtual plant model G_{HI} with its states and events aggregated by the information channel, which is not just a simple aggregation of the lower tier plant model.

One of the major theoretical issues in the above control architecture is the consistency of hierarchical control. This requires formulation of an analytical relationship between: (i) the required (closed loop) behavior resulting from the high level controller C_{HI} and high level virtual plant model G_{HI}; and (ii) the actual behavior implemented by the lower level controller C_{LO} over G_{LO}. The desired objective is that the virtual control is matched by the actual behavior of the executed control actions. Another important issue is identification and design of the information channel for implementation of the supervisory controller.

The following issues need to considered for future research in the development of supervisory controller of air operations in the discrete-event setting:

1. *Extension of the supervisory control system for a group of different types of platforms (e.g., aircraft) instead of identical platforms.* This extension will allow simultaneous operation of different types of aircraft and supporting weapon systems.
2. *Extension of the supervisory control system for operations under partial observability instead of complete observability.* This extension will allow operation where certain events may occur out of the supervisor's knowledge or be missing because of the communication failure.
3. *Extension of the supervisory control system for dynamic reconfiguration of the hierarchical structure.* This extension will enhance the flexibility of the supervisory controller under different types of combat operations such as re-deployment of idle aircraft or replenishment of lost aircraft.

Appendix A: Mathematical Preliminaries

A discrete event system is a dynamic system in which state changes are driven by instantaneous occurrences of events. Following the framework of Ramadge and Wonham [2], the discrete-event system to be controlled, called a *plant*, is modeled by a deterministic trim automaton

$$G = \langle Q, \Sigma, \delta, q_0, Q_m \rangle$$

where Σ is a finite alphabet of event labels, Q is a set of states, $q_0 \in Q$ is the starting state, $Q_m \subseteq Q$ is the set of marked states, and δ: $Q \times \Sigma \rightarrow Q$ is the (partial) transition function. The transition function is extended from event to trace δ: $Q \times \Sigma^* \rightarrow Q$ in the natural way, where Σ^* denotes the set of all finite length of event sequences over Σ including the empty string ε. The language generated by G is used to describe the closed behavior of the plant at the logical level. Formally,

$$L(G) = \{s \in \Sigma^* \mid \delta(q_0, s) \in Q\} \subseteq \Sigma^*$$

and the marked behavior

$$L_m(G) = \{s \in \Sigma^* \mid \delta(q_0, s) \in Q_m\} \subseteq L(G)$$

Since the marked states Q_m represent states of satisfactory completion and G is restricted to be trim, hence G is non-blocking, *i.e.*, every sequence generated by G can be extended to a state $p \in Q_m$. Formally, $pr(L_m(G)) = L(G)$ where $pr(L_m(G)) = \{s \in \Sigma^* \mid \exists t \in \Sigma^*, st \in L_m(G)\}$.

To impose supervision on the plant, the event set Σ is partitioned into two subsets Σ_c and Σ_u of controllable and uncontrollable events respectively, where $\Sigma_c \cap \Sigma_u = \varnothing$. A (centralized) supervisor is defined as a map $\gamma: L(G) \rightarrow 2^{\Sigma c}$, where $2^{\Sigma c}$ is the power set of Σ_c. The supervisor operates as follows: for each generated sequence of events $s \in L(G)$, the set $\gamma(s)$ consists of controllable events that are disabled by a supervisor γ after the occurrence of s. The closed-loop behavior of the system is represented by a FSA $(\gamma \| G)$. The language generated by the controlled system, denoted by $L(\gamma \| G)$, is defined as follows:

- $\varepsilon \in L(\gamma \| G)$
- $\forall s \in L(\gamma \| \mathrm{G})$ and $\forall \sigma \in \Sigma$,
  $s\sigma \in L(\gamma \| \mathrm{G}) \Leftrightarrow [s\sigma \in L(G)] \wedge [\sigma \notin \gamma(s)]$

A supervisory control γ is non-blocking with respect to plant G if $pr(L_m(\gamma \| G)) = L(\gamma \| G)$, in other words, the closed-loop system is non-blocking. Given a control specification, we first convert it into a prefix-closed language defined on the event set Σ, If the pre-specified specification language K is shown to be uncontrollable, we can compute the supremal controllable sub-language of K because the class of controllable sub-languages of K is closed under set union and has a unique supremum under set inclusion.

Synchronous Composition of two FSAs is used to represent concurrent operation of component systems, such as a squadron of several aircraft. Given two FSAs $G_1 = (Q_1, \Sigma_1, \delta_1, q_{01}, Q_{m1})$ and $\mathrm{G}_2 = (Q_2, \Sigma_2, \delta_2, q_{02}, Q_{m2})$ their *synchronous composition*, denoted as: $G_1 \| G_2 = (Q, \Sigma, \delta, q_0, Q_m)$, is defined as:

$$\delta(q,\sigma) = \begin{cases} (\delta_1(q_1,\sigma), \delta_2(q_2,\sigma)) & \\ \quad \text{if } \delta_1(q_1,\sigma), \delta_2(q_2,\sigma) \text{ defined } \sigma \in \Sigma_1 \cap \Sigma_2 & \\ (\delta_1(q_1,\sigma), q_2) & \\ \quad \text{if } \delta_1(q_1,\sigma) \text{ defined, } \sigma \in \Sigma_1 - \Sigma_2 & \\ (q_1, \delta_1(q_1,\sigma)) & \\ \quad \text{if } \delta_2(q_2,\sigma) \text{ defined, } \sigma \in \Sigma_2 - \Sigma_1 & \\ \text{undefined} & \\ \quad \text{otherwise} & \end{cases}$$

where $Q = Q_1 \times Q_2$; $\Sigma = \Sigma_1 \bigcup \Sigma_2$; $q_0 = (q_{01}, q_{02})$; $Q_m = Q_{m,1} \cap Q_{m,2}$; and $\forall q = (q_1, q_2) \in Q, \sigma \in \Sigma$. Thus, if an event belongs to the common event set $\Sigma_1 \bigcap \Sigma_2$, then it occurs synchronously in the two systems; otherwise, it occurs asynchronously.

Acknowledgements and disclaimers

This effort is sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-99-1-0547 (JFACC). The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

References

[1] P.E. Caines and Y.J. Wei. "The hierarchical lattices of a finite machine". *Systems Control Letters*, 25:257-263, July 1995.

[2] P.J. Ramadge and W.M. Wonham. "Supervisory control of a class of discrete-event processes". *SIAM J. Contr. Optim.*, Vol. 25, No. 1, pp.206-230, 1987.

[3] K. Rudie and W.M. Wonham. "Think globally, act locally: decentralized supervisory control". *IEEE Transactions on Automatic Control*, Vol. 37, No. 7, pp.1692-1708, Nov. 1999.

[4] K.C. Wong and W.M. Wonham. "Hierarchical control of discrete-event systems" *Discrete Event Dynamic Systems: Theory and Applications*, 6:241-273, July 1996.

[5] K.C. Wong and W.M. Wonham. "Modular control and coordination of discrete-event systems" *Discrete Event Dynamic Systems: Theory and Applications*, 6:241-273, July 1998.

[6] W.M. Wonham. *Notes on Control of Discrete-Event Systems. Systems Control Group, Dept. of Electrical and Computer Engineering*, U. of Toronto, April 1999.

[7] H. Zhong and W.M. Wonham. "On the consistency of hierarchical supervision in discrete event systems". *IEEE Transactions on Automatic Control*, Vol. 35, No. 10, pp 1125-1134, October 1990.

ENTERPRISE ARCHITECTURE ANALYSIS USING AN ARCHITECTURE DESCRIPTION LANGUAGE

Saurabh Sircar & Alexander Kott
Advanced Decision Support
Logica Carnegie Group
Pittsburgh, PA 15222 U.S.A.
{sircars, kotta}@logica.com

Abstract

The business of systems architecture mainly entails specification of components and links among these components. This paper augments the notion of systems architecture work by introducing specification of interaction protocols among the components. In such a situation, the idea of architecture analysis is also to make sure that the protocol employed among these component is also error-free in that there are no deadlocks, logical inconsistencies or undesired states of interactions. The central idea of this paper is to address such concerns by introducing the notion of Architecture Description Language (ADL) and using that as a formal way to specify interaction protocols. With a formal description of the system in ADL and using model-checking techniques in conjunction, the paper makes a case to augment techniques to analyze system architectures. Model checking may be used to detect and possibly correct errors and undesired states. The paper provides results of some simple experiments performed to prove the idea and makes some conclusions about the viability of such analyses when designing architectures for systems. In particular, a broker-based system is formally represented using an ADL and it is analyzed using model-checking tools.

1. Introduction

The motivation for model-checking techniques in architectural analyses stems from the fact that in modern enterprises, we see large-scale distribution of processes running concurrently and interacting in active or reactive manner and that there are no standardized techniques for their analyses. Often, some of these interactions are loosely defined and that lead to wasted time cycles in resources, high latency and lack of coordination. Generally, these problems can be ascribed to problems in interaction protocols among the various participants or components within the system. Errors creep into these protocols unwittingly simply because of the huge complexity of interactions and behaviors arising from temporal dependencies of participating processes. In trying to remove these errors, the first important step is to be able to model them at such an abstraction that these errors can actually be detected. We may then proceed to remove the problems by suitably modifying the protocols. With this emphasis on interaction protocols, we augment the realm of system architecture work. The next section describes this further.

2. System Architecture

In trying to do any work in systems architecture, it is imperative to know what we mean by it. There appear to be several related definitions of systems architecture. Depending on the type of approach one has towards systems architecture, its definition and the scope changes. In all cases, however, the general notion of system architecture entails specifying the *components* and the *links* among these components. A more detailed scrutiny of literature suggests that the business of system architecture is much more than just specifying the components and their interconnections or linkages [Rechtin97]. The idea of systems architecture carries with it the following facets:

- An underlying scheme (functions required) to effect actions
- Participating units (agents / entities / components) and their functions
- Links (channels / connections) among units
- Interfaces of the units

- Protocols of interaction among units

  Example: Consider information transfer between two computers. The underlying scheme could be that of file transfer (information is available only in files, say): the participating units are a client and a server; the link is an Ethernet cable connecting the two hosts (possibly via multiple hosts); interfaces are ports through which data is transferred; the protocol is the famous FTP protocol. All these taken together will comprise the system architecture for information transfer.

In this paper, we concentrate on minimal specification of components, links and interfaces for an example system and concentrate mainly on investigating the interaction protocols – the last facet listed above. To do this, we first describe what exactly is an ADL.

2.1. ADL

Architecture Description Languages or ADLs allow formal descriptions of system architectures. They provide clear and unambiguous syntax and semantics to describe processes, channels, components, ports, interface and protocols of interaction within the system. Like a programming language for software, an ADL allows these descriptions to be compiled and generate executable code for simulation and to verify certain types of system properties [GMW97], [BHMV97], [Holzmann91].

Within enterprise control, systems analyses using ADLs can play a very important role [JJV97]. This paper highlights the technique and shows, by means of an example of an agent-based system, the kinds of analyses that can be done. The efficacy of the technique is in being able to create error free designs, especially in situations where the enterprise consists of several agents that are interacting concurrently and asynchronously.

2.2. Technique and Analyses

Figure 1 below, shows the gist of the ADL technique. It is really an approach that is taken by most of the model-checking tools.

Theoretical underpinnings of model checking lie in temporal logic and timed automata [Holzmann97], [Alur94]. Several commercial and academic model-checking tools exist: KRONOS [Daws95] is an example of a commercial model-checker and HyTech [Henzinger97] and SPIN [Holzmann97] are promising research tools among many.

A formal description of the system in an ADL lets us study properties such as presence of deadlocks among two or more processes, reaching undesired states, unspecified receptions of messages among processes, unwarranted assumptions about process speeds or presence of any race conditions that have been introduced in the design. These are essentially the kinds of properties that are verified using model checkers [JMMS98] and it is indeed these kinds of analyses that will help us characterize and compare different architectures for an enterprise.

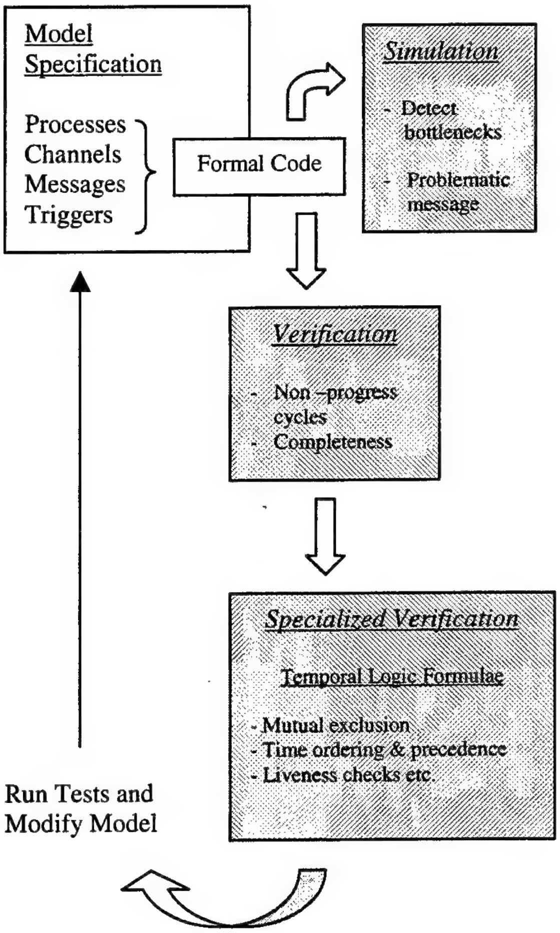


Figure 1. Gist of the ADL Method

As shown in Figure 1 above, there are two modes in which the models may be run: *simulation mode* and *verification mode*. In the simulation mode, a random seed is provided and the model is run to take a trajectory through state changes. In the verification mode, a certain formula may be specified to make claims about deadlocks or undesired states and then the model is run to verify if these situations may crop up in any trajectory through state changes that the system may take.

3. PROMELA, SPIN and Testbed Studio

In the analysis of hypothetical enterprises and architecture models of those enterprises, we used a language called Process Meta Language (PROMELA). This language allows us to write system descriptions that involve multiple, asynchronous, concurrent and distributed processes that communicate with each other through communication channels. The descriptions are then fed to a tool called SPIN to determine certain properties of the system. So, SPIN is a verification tool that takes system and process descriptions in PROMELA [Holzmann97]. We have used this tool to verify architectural properties of the system and thus, in essence, have used PROMELA as an ADL.

SPIN allows graphical analyses and simple user interfaces that make it easy to write system model descriptions in PROMELA.

A companion tool called Testbed Studio was also used to create similar models for a hypothetical enterprise. Testbed Studio provides a graphical interface to create models where the main entities are the processes and their actions. The underlying engine in Testbed Studio is actually SPIN [JMMFD98]. The graphical models in the tools are transparently rendered into PROMELA code and verification tests may be run guided by a graphical user interface.

4. Example Model

To illustrate the ideas of ADL-based architecture analyses, an example is studied and presented. In what follows, we give a specification of an agent-based system. The architectural specification of components and their interfaces are described in some detail. There is a brief description of the underlying scheme in the architecture – of brokering agents – and, an idea of the information distribution patterns. The agents are set in a closed loop framework, which means that there is a mechanism of feedback on actions and effects sent to some agents that are responsible for overall direction of the enterprise's actions.

4.1. Closed Loop Framework for an Agent Based Enterprise

Inspired by mechanisms within a market, a scheme of operations is conceived. In the example, we develop architectures over such a scheme and then perform ADL-based analyses. The idea of such broker-based (B2) enterprise was first described in [LCG00] for a military command and control domain. It has been modified in this paper to a generic enterprise. The main theme is that it employs a notion of agents that bid and compete for a task or a job and try to get it done. A few high-level details of participating agents follow:

Objective Leaders (OL) initiate processes. Each OL has one or more objectives (could be set by a higher-level authority). They formulate jobs that need to be done to meet the objectives. An OL calls for bids to get those jobs done; it may be given a budget of "attractors" (dollars, brownie points, rewards etc.) as incentives and to attract bids.

Job Brokers (JB) get paid for matching jobs with job seekers. JBs know the job market, their clientele, their capabilities, and the current situation; they compete with each other and promote jobs to suitable job seekers.

Entrepreneurs (E) look for complex jobs that require coordination of multiple Workers. Es bid for those jobs and issue sub-bids to Workers; they get paid for getting jobs done. Es may not bid in all cases but in most instances, they compete for jobs.

Workers (W) get paid for getting jobs done. Like Es, they also compete for jobs. Typically, an E gathers a team of Ws to get a job done.

Situation Brokers (SB) listen to all traffic and perform analyses of information. They notice conflicts, problems, threats and opportunities, and issue tips or alerts to OLs, JBs and Es. SBs compete for and get paid by subscribers (OL, JBs and Es).

It is instructive to note that all these agents perform vital functions within an enterprise and to effect a control framework, there is a built-in feedback mechanism through SBs. The quality of feedback will depend upon a deal that transpires between the SBs and their subscribers. The OLs may induce others "attractors" to improve the quality of feedback.

An interesting point to observe here is that the scheme permits multiple levels of feedback (information sent to OLs is different from that sent to Es or JBs, for instance). Like in a standard distributed control framework, this scheme also has possibilities for differences in state estimation, regulation functions and observations. However, in this study of a B2 enterprise, we concentrate on message exchanges among the brokers and agents and search for inefficient or harmful dependencies that may be present.

4.2. Enterprise Architectures over a B2 Scheme

In order to create architectures over the B2 scheme described above, we impose certain structural components for the enterprise. The most important of such structures is that of communication mechanisms which provide coupling among the various agents. For instance, a Market Blackboard (MB) allows for a shared resource for communication; a full-mesh connectivity with dedicated channels provides another way of linking the agents. An important observation is that there are many functions implicit in the B2 scheme, which require additional constructs and structures. It is precisely the arrangement of these structures that lead to variants of enterprise architectures that we wish to analyze. We delineate below, some of these primary functions that could lead to different architectures:

- Receiving calls for bids from OLs and notifications to JBs, Es and Ws.
- Receiving special bid promotions from JBs and making them available to Es and Ws.
- Receiving and storing bids from JBs, Es and Ws.
- Selecting winning bids, and informing winners and losers (JBs, Es and Ws).
- Receiving a stock of "attractors" from the OLs to use for "payments" to the other participants.
- Receiving advertisements for subscriptions from SBs and making them available to other participants, who may respond with requests for subscriptions.
- Relaying requests for subscriptions to SBs from OLs, JB, Es and Ws.
- Issuing payments to SBs in association with subscription requests from others.
- Issuing payments to JBs in association with bid wins.
- Issuing payments to Es and Ws for successful completion of jobs, based on evidence provided in situation reports.
- Collecting situation reports and problem reports from Es and Ws and relaying them to OLs (and also SBs, who receive all information that flows).

The value of ADL-based models for enterprise architectures is to see the efficiency of protocols that can be specified for different architectures over a general scheme of brokering agents. Efficiency in such a case is the absence of deadlocks and unwanted states. The other value is in being able to see which variant of architecture performs better over the B2 scheme.

5. Experiments and Results

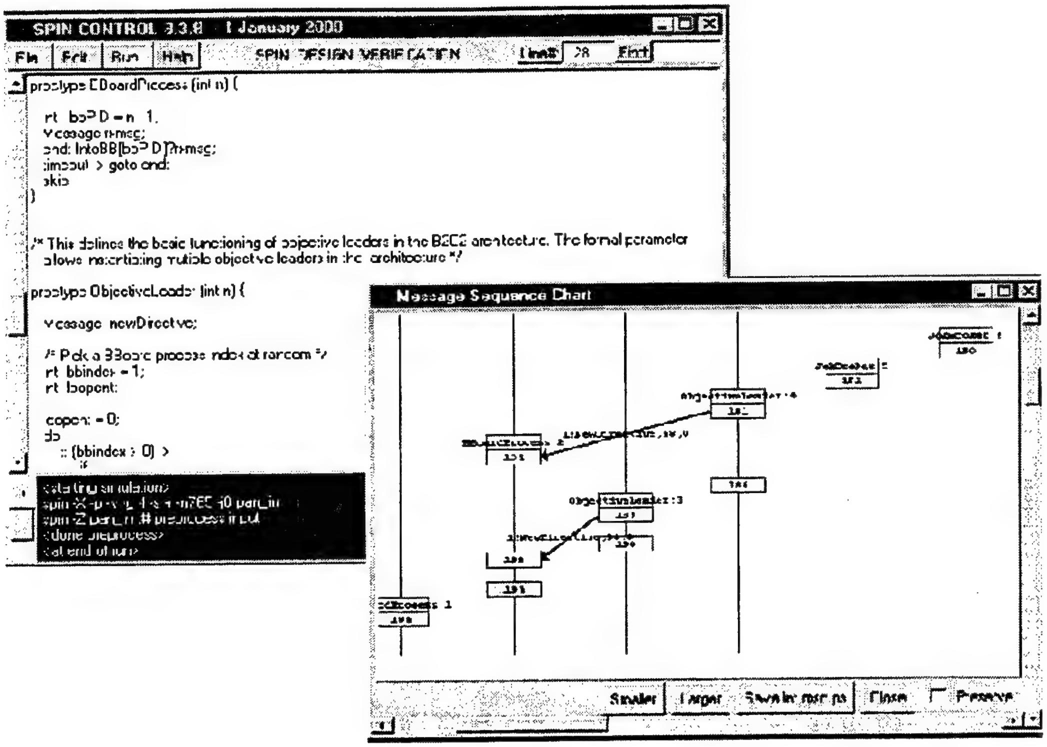
We have designed a few experiments with two architectural models based on communication mechanisms imposed over the B2 scheme.

5.1. PROMELA Models

With models created using PROMELA and run directly with the SPIN tool, we tested two possible communication schemes: in one, we effect all communication via a MB (as described above) with full-mesh connectivity and in the other, the MB is in the form of a ring.

In another experiment, we determine the effect of bid withdrawals on the state of the task assigned by the OLs.

Linear Temporal Logic (LTL) formulae were created within the SPIN tool to run verification



```
proctype BBoardProcess (int n) {

  int  bbID = n - 1;
  Message r-msg;
  end: IntoBB[bbID]?r-msg;
  timeout -> goto end;
  skip
}

/* This defines the basic functioning of objective leaders in the B2C2 architecture. The formal parameter
  allows instantiating multiple objective leaders in the architecture */

proctype ObjectiveLeader (int n) {

  Message newDirective;

  /* Pick a BBoard process index at random */
  int  bbindex = 1;
  int  loopcnt;

  loopcnt = 0;
  do
     :: (bbindex > 0) ->
         if
```

```
<starting simulation>
spin -X -p -v -g -l -s -r -n7EC -j0 pan_in
spin -Z pan_in # preprocess input
<done preprocess>
<at end of run>
```

Figure 2. SPIN windows showing PROMELA code for B2 Scheme and message sequence for a simulation run

tests. The formulae have been parsed and interpreted here for the benefit of the reader. The key to creating LTL formulae is to looks for situations where there is parallelism of processes or' there are split-join situations among processes. Figure 2 shows screen captures of PROMELA code and message sequence for a ring structures MB.

5.1.1. Experiment 1

Efficiency of two distributed market blackboard (MB) schemes: Compare messaging volumes for the two different schemes.

Architecture features:
- Full mesh structure using 4 MB processes.
- Ring structure using 4 MB processes.

Verifications using LTL formulae:

([] fullMeshMsgCount > 4) :

For every message, a full mesh structure for the distributed MB always generates more than 4 copies – TRUE

([] ringMsgCount == 3) :

For every message, a ring structure for the distributed MB always generates exactly 3 copies – TRUE

Result: Message volumes in the full mesh structure always exceeded those in the ring structure for the distributed MB.

5.1.2. Experiment 2

Detect pathologies of possible indefinite waits: Determine if bid withdrawals by agents may render an assigned task in an indeterminate state.

Protocol features:
- A JB may withdraw a bid anytime.
- An E may withdraw a sub-bid anytime.

- Live states of an assigned task are when it is either being bid or when it is actually executing.

Verification using LTL formula:

[] bidWithdraw ->
(taskState==processing || taskState==executing):
Is it always true that a bid withdrawal will lead to the state of the task as being processed or being executed? – FALSE

Result: Bid withdrawal may cause indeterminate state of the assigned task.

5.1.3. General Conclusions

Matters to fix in communication links:

- Create a mix of distributed BB and dedicated channels for communication among agents.

Matters to fix in the present protocol:

- The B2 scheme requires the feedback protocol to guarantee that there is at least one SB that wins a contract or the OLs have to maintain dedicated feedback channels of their own.
- Bid withdrawals may not be allowed in the protocol after OLs have published winners.

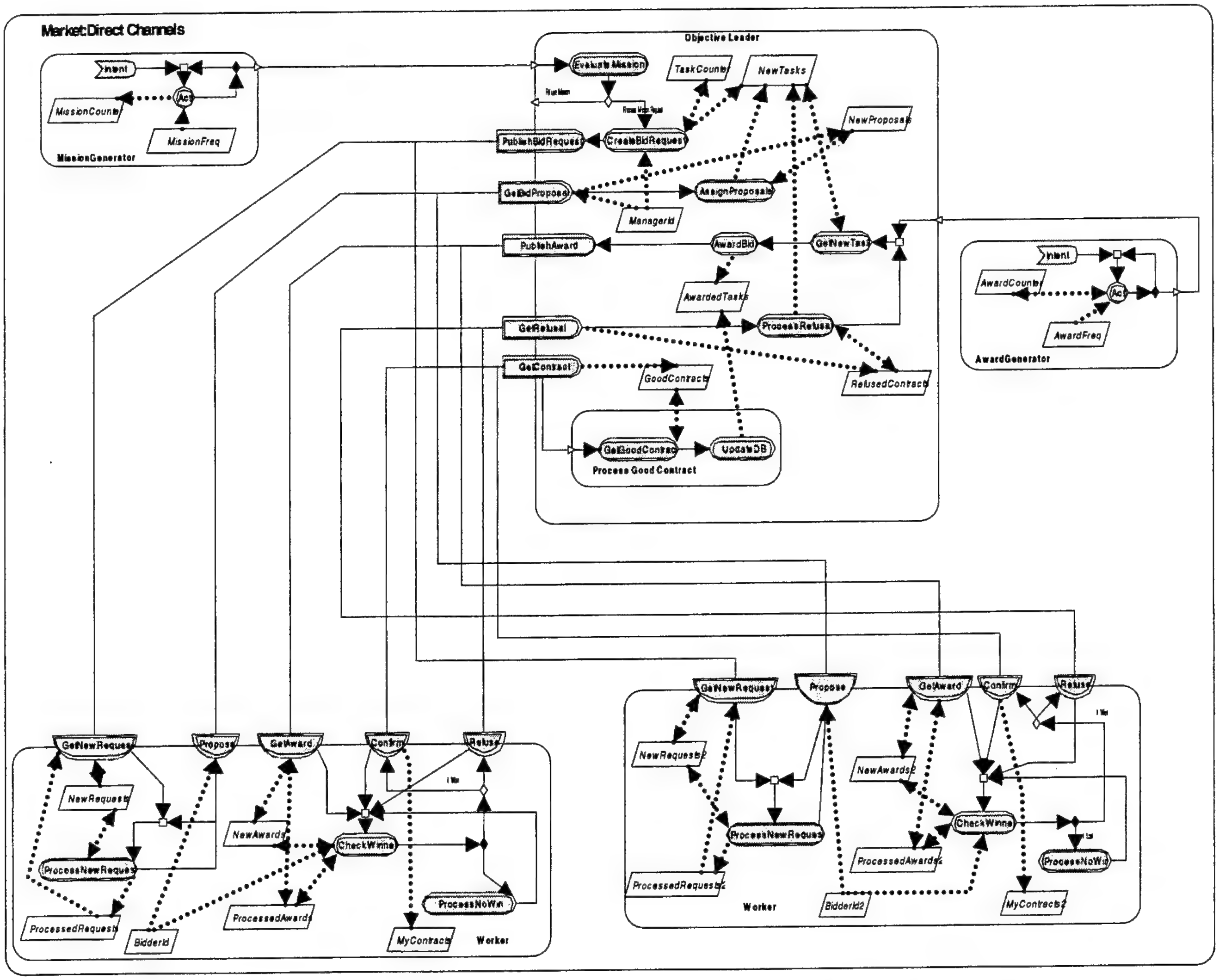


Figure 3. Testbed Studio Model for a Simplified Broker-Based Scheme

5.2. Testbed Studio Models

Creating LTL formulae to verify the architectural properties of a system is not always an easy task. Testbed Studio is a tool that allows creating architectural models graphically and further, alleviates the burden of creating LTL formulae. In doing analyses with Testbed Studio, four different *patterns* of functional verification [JMMFD98] are possible:

- *Tracing pattern*: Action X occurs {always, ever, never}
- *Consequence Pattern (Liveness):* Action X leads to Action Y
- *Combined Occurrence:* Actions X and Y occur {mutually exclusive, together always, together ever}
- *Precedence (Safety):* Actions from set X require Actions from set Y.

Two models that were created with this tool are variations of those created with PROMELA. The number of agents implemented in these models were considerably less because of computational constraints.

In both models, we have an OL that has many sub-functions (indicative of the actions it would take to effect the protocol) like evaluating missions, assigning proposals, evaluating and publishing bids, deciding on bid winners and task assignments. There are two Ws (bidders) bidding for tasks. The Ws also have sub-functions of getting new requests, processing of the requests, proposing (bidding), confirming task assignment, checking win status etc.

The difference between the two models is that in one, all communication are through the MB and in the other, there is a mix of MB and direct communication channels among the OL and the Ws.

Separate event triggers have been implemented in both the models to start the process of publishing and retrieving bid requests and award generation.

Figure 3 above, shows one of the models. The other model is very similar in terms of functional blocks; only the communication channels are different.

5.2.1. Experiments

The experiments with the Tesbed Studio models for the B2 scheme were mainly verification tests based on the four patterns listed before. The tests render a true / false answer to the questions asked from the model.

Some Results:

1. Liveness tests:

- Award Bid action in the OL is *always* executed – FALSE.
- Award Bid action in OL is *ever* executed – TRUE.

2. Precedence tests:

- Check Winner action in a W is *always preceded* by its Propose action – TRUE.

3. Combined Occurrence:

- Award Bid and Assign Proposal actions in OL are *never* executed together – TRUE.

6. Conclusions

The main thrusts of the paper are the following: first, superposed over a closed-loop framework, the use of model checking techniques allows an enterprise architect to test the proposed framework for logical errors and inefficiencies. If inadequacies are observed, changes in the control framework or rearrangement of interaction sequences may be introduced. Thus, any such changes are introduced by means of logical and formal analyses rather than by intuition. Second, we report some results of experiments performed with two different architectural models for a new scheme of interacting agents based on market mechanisms. The results help us to see some merits and demerits of the architectures. Performing the comparisons allows meaningful tradeoffs toward architecture design.

7. Acknowledgements

The research was funded in part by a grant from DARPA on the JFACC project under contract number MDA972-00-C-0034.

The concepts of a broker-based scheme for an enterprise were guided by ideas generated at Logica Carnegie Group. Technical staff in the JFACC project had a large part in contributing and developing some of the key ideas. Special thanks go to Alexander Slavkovic for creating the models in Testbed Studio and helping with verification tests and analyses.

8. References

[Rechtin97] E.Rechtin and M.Maier, *The Art of Systems Architecting*, CRC Press, 1997.

[BHMV97] A. Basu, M. Hayden, G. Morrisett, T. von Eicken, A Language-Based Approach to Protocol Construction, Proceedings of ACM SIGPLAN Workshop Domain Specific Languages (WDSL), Paris, France, Jan. 1997.

[GMW97] David Garlan, Robert Monroe, David Wile, Acme: An Architecture Description Language, Proceedings of CASCON'97, Nov. 1997.

[Holzmann91] Gerard Holzmann, *Design and Validation of Computer Protocols,* Prentice Hall, 1991.

[Holzmann97] Gerard Holzmann, The Model Checker SPIN, IEEE Transactions on Software Engineering, Vol. 23, No. 5, pp. 279-295, May 1997.

[Alur94] R. Alur and D.L.Dill, A Theory of Timed Automata, *Theoretical Computer Science,* 126:183-235, 1994.

[Daws95] C. Daws and S.Yovine, Two examples of verification of multirate timed automata with KRONOS, Proc. IEEE Real-Time Systems Symposium, Pisa, Italy, Dec.1995.

[Henzinger97] T.A.Henzinger, P.H.Ho and H.Wong-Toi, HyTech: A Model Checker for Hybrid Systems, Software Tools for Technology Transfer 1:110-122, 1997.

[JJV97] W. Janssen, H. Jonkers, J. Verhoosel, What Makes Business Processes Special? - An Evaluation Framework for Modeling Languages in Business Process Redesign, Proceedings of the 2nd CAiSE/IFIP 8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design, Barcelona, Spain, 1997.

[JMMS98] W. Janssen, R. Mateescu, S. Mauw, J. Springintveld, Verifying Business Processes Using SPIN, Proceedings of the 4th International SPIN Workshop, Paris, France, Nov. 1998.

[JMMFD98] W.Janssen, R.Mattescu, S.Mauw, P.Fennema, P. der Stappen, Model Checking for Managers, Proceedings of the 4th International SPIN Workshop, Paris, France, Nov. 1998.

[LCG00] Logica Carnegie Group, JFACC Architecture Design Repository, March 2000.

Closed-Loop Operation of Large-Scale Enterprises: Application of a Decomposition Approach

Corresponding Author:
Milton B. Adams
C.S Draper Laboratory, MS # 3F
555 Technology Square
Cambridge, MA 02139
Ph: 617-258-3185
Fax: 617-258-1799
adamsm@draper.com

All other authors also with Draper Lab:
Owen L. Deutsch
William D. Hall
Richard H. Hildebrant
William R. Kreamer
Marc W. McConley
Hon Fai Vuong

1 Abstract

Real-time, closed-loop optimal control of large-scale dynamic systems (enterprises) remains a challenging problem [6]. We have been developing an approach to problems of this class that employs a distributed, multi-level control architecture wherein planning and execution are decomposed to accommodate the near and far term impacts of plant disturbances and modeling uncertainties. The decomposition is based on the theory of multi-level optimization for large-scale systems [4, etc.]. The structure of the decomposed solution to the optimization problem obtained from this theory forms a basis for our controller architecture as well. In addition to planning, the controller architecture includes execution management, monitoring and diagnosis at each level. A previous paper described a decomposed formulation for a large-scale military air operations optimization problem [1]. This paper presents the results of the application of this approach to the control of large scale military air operations in a simulation-based context. Simulation results indicate that a significant reduction in the time required to achieve specified campaign objectives can be realized by closing the control loop at higher rates facilitated by controller automation. This reduction pertains to the base case and to cases with modeling errors and disturbances and can be quantified as a savings of 0.5 to 2 days for the moderate intensity, 7 day scenario under study.

2 Overview

Military air operations require command and control of diverse forces distributed over large geographic areas. The geographic distribution coupled with the need for short decision cycle times requires an agile, distributed and collaborative command and control capability for effective dynamic tasking of strike packages, supporting logistics, and sensing and electronic warfare assets.

This paper describes an approach to decomposing and executing large-scale decision-making problems for dynamic environments that combines the theories of decomposition of large-scale optimization problems and distributed control. This enables the replacement of heretofore ad hoc approaches to decomposing this class of large-scale operational problems, resulting in a distributed system for which the problem-solving and decision-making within each distributed C^2node addresses enterprise-wide objectives. Employing this approach to decomposition both provides significant insight into the nature of the feedback required to close the loop around each of the control nodes within the decomposed problem, and defines the dynamics of the interactions among the control nodes in solving the enterprise-wide problem, including the objectives passed from superior nodes to subordinate nodes and the feedback/status passed from subordinates to superiors.

3 Technical Approach

Our previous paper [1] described in detail the problem formulation, approach to decomposition and the controller architecture. For completeness, an overview is provided here as well. The focus of this paper is on the description of the solutions we have developed for that formulation and the outcomes of the experiments that we have performed in evaluating our implementation of the solution.

3.1 Decomposition and Closed-Loop Control

The theory of large-scale optimization [2,3,4,5] provides a variety of approaches to decomposition of very large scale problems into components or subproblems that are computationally tractable. The objective of multi-level optimization is to decompose a complex optimization problem into a hierarchy of simpler problems. The simpler optimization problems are solved independently at each level of the hierarchy, with the superior or master levels coordinating the solutions of the decoupled subordinate level problems.

The following basic questions must be answered in developing decompositions for large-scale enterprise operations:

- How many levels are required?
- What constraints and objectives should be passed from level to level?
- What is the nature of the status that is passed from subordinate to superior levels?
- How is problem-solving best accomplished across levels?
- What happens when a level cannot meet its objectives and/or honor its constraints?
- To what extent should the decomposition reflect human-system interaction concerns?
- How might one develop a system wherein levels are established dynamically?

The central topic of our effort is the extension of analytical approaches to decomposing large-scale optimization problems to closed-loop control for large-scale enterprise problems with complex objective functions and constraints. The solutions to the subproblems at the lowest levels of the decomposition represent a plan of activities that are to be pursued by the enterprise's physical entities in prosecuting the business of the enterprise, e.g., missions for individual aircraft. At higher levels, the solutions produce objectives and constraints to be employed by successively lower levels, e.g., allocation of sets of targets to sets of strike packages. The environment (the plant) within which those activities are to be pursued is represented (modeled) in the formulation through a variety of constraints. In order to reduce sensitivity to disturbances and modeling errors, we employ feedback providing information about the actual evolution of the state. The following describes the control architecture that we use in closing the loop.

Figure 1 represents one of the command and control nodes within the controller architecture. Feedback is provided by sensing the "system to be controlled." The "system" may be physical entities within the plant that are being controlled or it may represent an aggregation of lower level problem solving nodes along with the entities they control. A closed-loop, hierarchical decomposition is a recursive implementation of the node illustrated in Figure 1, where the "system-to-be-controlled" is one or more subordinate level processes that are "controlled" or coordinated by an upper Master level as shown in Figure 2.

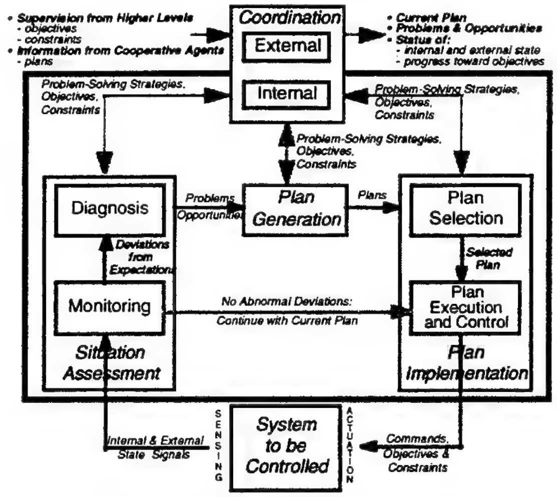

• Supervision from Higher Levels
- objectives
- constraints
• Information from Cooperative Agents
- plans
Coordination
External
Internal
• Current Plan
• Problems & Opportunities
• Status of:
- internal and external state
- progress toward objectives
Problem-Solving Strategies, Objectives, Constraints
Problem-Solving Strategies, Objectives, Constraints
Problem-Solving Strategies, Objectives, Constraints
Diagnosis
Problems
Opportunities
Plan Generation
Plans
Plan Selection
Deviations from Expectations
Selected Plan
Monitoring
No Abnormal Deviations:
Continue with Current Plan
Plan Execution and Control
Situation Assessment
Plan Implementation
Internal & External State Signals
SENSING
System to be Controlled
ACTUATION
Commands, Objectives & Constraints

Figure 1: Functional Decomposition of a Command and Control Node

In Figure 2, the plant for the Master level is an aggregation of the lower level nodes and their plants. For the results discussed here, the lowest levels control a simulation rather than the actual plant.

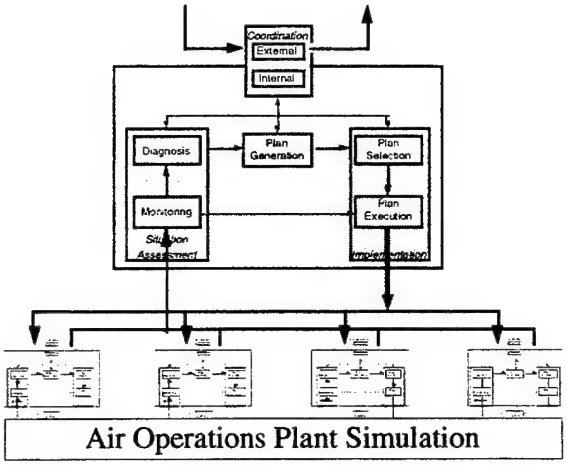


Figure 2: Hierarchical View: Aggregated Plant for Master Level

The nature of the feedback required is a function of the nature of the subproblems to be solved. The feedback should contain the information required to evaluate progress toward the solution to the subproblem being solved. Since the solutions generated will span a finite time horizon, models will be required to predict future states and status based on the planned course of action and estimates of the current state.

4 Problem Formulation & Decomposition

4.1 Strike Planning Problem

We address a strike planning problem wherein aircraft and weapons are tasked to strike targets of interest.

Each aircraft is based at one of a number of bases. Aircraft are tasked together in groups called strike packages. The aircraft in a strike package work together to accomplish a mission objective. Different aircraft in each strike package typically perform specialized functions that contribute to the overall effectiveness of the package. For instance, a strike package may consist of bombers escorted by fighters. The fighters in the strike package engage enemy air patrols that could hamper the bombers' ability to reach their targets. The components of a strike package that are appropriate for a given mission objective depend on the targets to be struck, the types and amounts of enemy resistance expected enroute and at the targets, and the competition for strike resources from other mission objectives.

The objective of the strike planning problem is to assign resources from the bases to the targets in a way that maximizes the expected accomplishment of mission objectives defined by the commander's intent. We have constrained the problem so that no more than one strike package strikes a target in any plan instance, although it is sometimes necessary to repeat attacks against targets at later times. The details of the mathematical formulation and decomposition can be found in [1].

To summarize, the scope of the air operations planning problem that we address includes:

- Deciding which targets to hit when
- Deciding which assets to use to deliver weapons
- Deciding routes to follow, refueling and assembly points
- Assigning wild weasel and jammer escorts
- Respecting laws of physics, logistics and human performance
- Accounting for risk in decisions

The major assumptions include:

- Campaign objectives express commander's guidance and are used to drive possible courses of action
- State estimates are provided to the controller at regular intervals

4.2 Decompositions

Consider a decomposition of the strike planning problem in which an upper level C^2 node assigns targets to bases and each base is responsible for determining the strike packages to assign to each target. The decomposition comprises a master problem and a series of subproblems. Each subproblem is associated with a base, and the master problem is associated with the upper level C^2 node. The base problems cannot be independently solved without coordination from the upper level because, without coordination, multiple bases would likely plan to strike the same high valued targets, violating the constraint that each target be struck by no more than one strike package. The upper level coordination addresses these issues, ensuring that some targets will not be ignored due to resources being wasted on the multiply-hit target.

4.3 Commander's Intent Based Planning

Air operations plans must be driven by commander's intent. Commander's intent reflects:

- ***Time***: Importance of Campaign Phase (time phase importance or target time criticality)
- ***Geography***: Importance of region
- ***Target Class***: Importance of target class or grouping of targets
- ***Risk***: Importance of achieving objectives vs. loss of resources (including human)

Commander's intent is captured in the "Commander's Intent Input Matrix." The information contained in the matrix is mapped into target values that vary with region, by time, by class - along with a tradeoff of the benefit of successfully prosecuting the target versus the cost of success. This approach to modeling commander's intent is intended to illustrate that commander's intent can be captured to drive the planning problem.

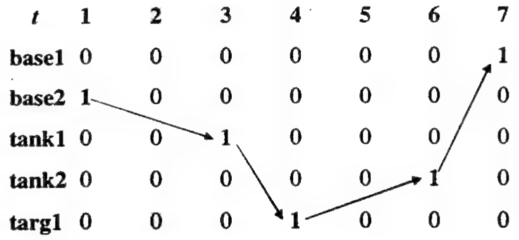
Modeled target value also expresses the notion that targets may have independent valuation as well as valuation as part of a "target system." Our proposed target value model includes the following components:

1. Commander's guidance expressed as a relative weighting between functional categories and geographic locations that may be a function of campaign phase.
2. Threshold of damage to individual targets before any contribution to plan valuation
3. Utilization of weaponeering inputs for probability of damage to particular targets for particular weaponeering selections (weapons and aimpoints).
4. Payoff functions that express collective effects on "target systems"
5. Time perishability factors for targets whose value or ability to strike is fleeting and for which reconstitution may occur.

The commander's guidance is concretely expressed as a table of weightings with rows varying over target functional categories and columns expressing geographic regions or groupings. There may be a number of tables expressing guidance inputs for each campaign phase. Given finite strike resources, the table expresses the relative value to the commander of achieving damage to targets in different slots. Instead of a rigid, top-down allocation of resources, this table is used to establish objective value for alternative strike plans. Targets in lower weighted slots may, in fact, be incrementally favored over other targets because of collective effects.

The payoff function is applied separately in each category/region slot and may be linear for targets with independent valuation or nonlinear for targets with collective valuation. The overall or aggregate value of all targets is formulated as the weighted sum of payoffs for each slot with weightings specified by the commander's guidance tables. It should be noted that the payoff metric can be evaluated with the current damage state of targets (i.e., strikes already executed) or with the projected damage state for current plans (i.e., including future planned strikes). The former is useful for monitoring actual target value accrued as a function of time as opposed to projected accruals including the effects of future planned strikes. Of course, the actual damage state achieved may be different than the projected damage state for any strike by virtue of simulated discrepancies in weaponeering estimation.

4.4 Decision Variables



t	1	2	3	4	5	6	7
base1	0	0	0	0	0	0	1
base2	1	0	0	0	0	0	0
tank1	0	0	1	0	0	0	0
tank2	0	0	0	0	0	1	0
targ1	0	0	0	1	0	0	0

Figure 3. Decision variables y_{ijt} ***for Aircraft i***

Plans are expressed as times of arrival of each aircraft at each location of interest as illustrated in Figure 3, where **locations of interest** ***j*** include bases, tanker orbits, targets and the current locations of all aircraft, The decision variables are binary with $y_{ijt} = 1$ if aircraft ***i*** is at or is to arrive at location ***j*** during **time interval** ***t***.

This decision variable structure is flexible in that: aircraft can assemble from different locations, routing to visit multiple targets is supported, enroute aircraft can be retasked and new packages formed. Indeed, almost any reasonable strike mission can be represented in this manner.

In formulating a solution, a variety of constraints are imposed in the development of strike mission plans. These include: time is required for aircraft to travel, fuel endurance must be respected, pilot/crew endurance must be respected, aircraft payload is constrained, weapons may only be replenished at bases, fuel may only be replenished at tankers or bases, and re-tasking in flight may incur additional time and risk.

The decision variable structure has been selected to reflect the level of detail required to represent a mission plan. However the mathematical complexity of developing optimal solutions to this constrained optimization problem for the case where there are hundreds of strikers and hundreds of targets is so high that it is computationally infeasible within the time constraints (less than 10 minutes) required to rapidly close the loop in response to unanticipated events. To resolve this problem, as discussed earlier, we have decomposed the problem into simpler problems that can be solved within these time constraints [1].

4.5 Decomposition Architecture

The levels of the decomposed strike planning problem are depicted in Figure 4. The Target Partitioning level (Level 1) assigns targets and aircraft and weapon resources to Mission Generation (Level 2) planners. The Level 2 planners assign and schedule specific aircraft and weapon resources for specific targets. The Mission Generation decision making is supported by the Strategic Router (Level 3) which provides attrition, time and fuel costs for specific target-aircraft pairs, accounting for availability of jammer and escorts and constrained by assembly points as specified by the Level 2 planner.

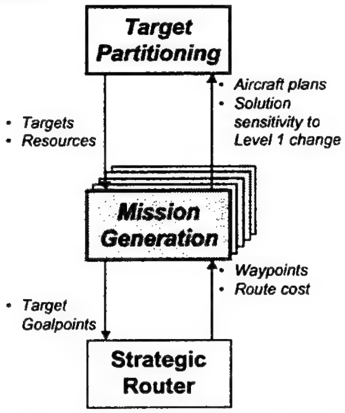


Figure 4. Three Level Planner Decomposition

Both optimal and heuristic solutions are being formulated to solve the Level 1 and Level 2 constrained optimization problems. The heuristic planners are useful for both development and decomposition experiments, with the expectation that the objective values produced by heuristic planners will follow trends similar to those of the optimal controller.

Indeed, we plan to use the heuristic solutions as part of the final controller. In particular, solving the decomposed problem requires several iterations (negotiation exchanges) between the Level 1 and Level 2 planners in order to approach optimality. Our approach is to have the initial iterations performed with heuristic Level 1 and Level 2 solutions, which are much faster to implement than the optimal. In later iterations, the optimal solutions will replace the heuristics. This can be viewed as providing a "warm start" for the optimal solutions.

The level 1 problem is to partition the list of available targets into disjoint sets and to assign one set to each Level 2 subproblem along with aircraft resources. For our initial implementations, we will preselect the resources to be allocated to the Level 2 planners (e.g., consider a Level 2 planner a base at a fixed geographic location with a pre-assigned set of aircraft).

4.5.1 Optimal Solution Methodology

Direct solution of the optimization problem formulated in [1] is computationally intensive. Indeed, its direct solution is intractable for realistic problem scenarios. Fortunately, the *composite variable* approach developed in [7] applies to this problem with encouraging results.

The composite variable approach produces an equivalent but stronger integer programming formulation through the translation of decision variables. In the translation, a new "composite" variable is defined for each feasible combination of certain sets of decision variables. In our application, setting a single composite variable to 1 is equivalent to appropriately setting the y_{ijt} for all aircraft i in a given package attacking a target, refueling at tankers, and returning to their respective bases. The resulting formulation is far stronger than that based directly on the y_{ijt}.

Of course, if all possible combinations of the y_{ijt} were enumerated as separate composite variables, the problem size would grow extremely large. Fortunately, many of the combinations of y_{ijt} that could potentially define composite variables are dominated by better choices. In order to create only the relatively very small set of composite variables that are not dominated by other composite variables, an optimization subproblem is solved. Results of implementation of this approach will appear in a future publication.

4.5.2 Heuristic Algorithms for Levels 1 - 3

The ***heuristic Target Partitioning (Level 1) planner*** allocates targets to the Mission Generation (Level 2) subproblems to encourage the generation of plans which prosecute high valued targets in a timely manner while maintaining workload balance among subproblems. A target list is created and ranked in order of decreasing target value. The Level 1 planner allocates each target on the target list to its nearest base, starting with the highest valued target, thereby allowing higher valued targets to be prosecuted sooner. Workload balance is maintained by ensuring the number of targets allocated to each base is proportional to that base's workload capacity. The workload capacity is measured as a function of that base's weapon delivery capacity.

The ***heuristic Mission Generation planner*** has been to maximize target value specified by region and functional category as established via the *Commander's Intent Input Matrix* while simultaneously attempting to minimize the total cost comprising:

- Attrition risk
- Cost of time
- Weapon utilization
- Mission re-tasking cost

The heuristic implementation of the Level 2 planner provides a baseline controller for comparison with experiments of the optimal integer programming solution. In addition, as described above it may be used in conjunction with the optimal planner to speed convergence by providing a warm start. It sequentially constructs strike packages, assigning aircraft to targets, optimizing the incremental contribution to an objective function for each aircraft mission that the planner generates. This incremental approach has been found to generate reasonable plans very quickly. The optimization accounts for:

- Target prioritization
- Assignment of aircraft to targets
- Weaponeering via pre-specified package configurations
- Asynchronous scheduling of sorties with package-level synchronization of time on target

The Level 3 planner is a ***Strategic Router*** that supports the Level 2 planner by providing the cost of constrained minimum risk routes for specified aircraft-target pairs.

The route planning problem is:

Given a set of

1. **Mission parameters** including:
   - Start location (base or enroute) and return base
   - Required ingress and/or egress assembly points
   - Target location
   - Set of all tanker locations
2. **Aircraft parameters** including:
   - fuel endurance,
   - pilot endurance
3. a **Threat model** including:
   - threat density,
   - detection range,
   - likelihood of engaging,
   - P_k
4. **Escort level** representing one of:

- No escort,
- wild weasels,
- weasels plus escort jammers

Determine a strategic-level route (10 - 30 km grid spacing for waypoints) that is:

- fuel feasible
- minimizes risk from threat engagement
- allows for two refueling activities on each segment
- adheres to specified assembly points

An A* search based router has been implemented and integrated with the Level 2 planner. Figure 5 illustrates the results from the router for a 30 *km* gridding with a required assembly point on ingress. The legend indicates base, tanker, target and assembly point locations. Threat levels are color coded from low (yellow) to red (high).

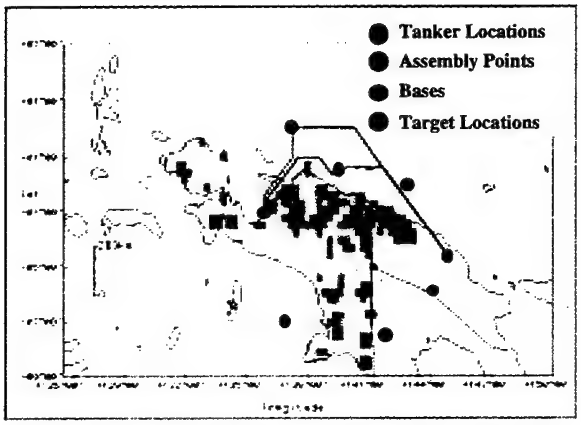


Figure 5 Example Routing with Required Assembly Point on Ingress

5 Experiments and Results

A set of experiments has been designed to investigate the air operations effectiveness that results from the application of our distributed control architectures. Specifically, these experiments investigate

- the improvements gained and sensitivity to the rate of loop closure,
- the robustness of our closed loop architecture to uncertainties in the models employed for blue aircraft attrition and weapons effectiveness and
- the agility of the close-loop system response to command level designation of new targets and/or changes in priority/valuation of existing targets.

Experimental results are obtained by applying our hierarchical controller to a simulation of a military air operation scenario, wherein only the salient characteristics of military air operations are modeled. The experiment scenario has been chosen to provide an operational setting wherein we can credibly illustrate that feedback and shorter control cycle times yield improved performance and robustness to modeling uncertainties and a changing operational environment. Randomization via small numbers of Monte Carlo trials has been employed for assessing the performance and robustness of our closed-loop controller design.

The state of the plant has been limited to blue (friendly) and red (adversary) forces, with the principal *blue* state elements being air and supporting resources including bases and weapons stores; and the principal *red* state elements air defense, targets and supporting components. The intent is to control the evolution of the blue state and influence the evolution of the red state.

The classes of plant disturbances include:

- unanticipated changes in blue resources due to unexpected rates of attrition;
- unanticipated discrepancy in effectiveness of weapons employed; and
- unanticipated changes in red activities reflected in changing target locations and values.

The primary metrics employed for the evaluation of experiments are those ***associated with the accomplishment of the commanders inten***t. Those metrics include the aggregate value of target destruction by (a) target category, (b) operational geographic region and (c) campaign phase (time) as well as (d) time to achieve levels of fractional destruction along these same dimensions. On the cost side, the attrition of aircraft and the cost of utilization of munitions and mission support resources is logged and included as an element of the evaluation for each experiment. The results presented here focus on the aggregate target damage metric.

In addition to cost and plan value, we also evaluate the performance of our closed loop controller in the context of "plan stability." Here, plan stability is a measure of how plans changes each time the loop is closed and new plans are developed. From a human factors perspective, it is unacceptable to have frequent, significant changes in strike plans for individual aircraft, especially when they are already en route to a target.

5.1 Experimental Results

The results of our experiments have provided strong support for the hypothesis that closed-loop solutions to realistic large-scale air operations planning problems can be largely automated and that the increased loop closure rates resulting from automation can substantially improve overall system performance. In the experimental results that follow, we focus on the difference in performance of closing the air operations planning and control loop at a low rate (24 hour cycle) and a relatively high rate (4 hour cycle). Four hour and shorter cycles are made possible by automation of the plan generation process as described earlier and improved battlefield information dissemination, providing the required higher rate of feedback.

Figure 6 is a segment of a screen snapshot from our simulation of the execution of the strike missions planned by our closed-loop controller. The locations of tankers, assembly points and bases are identical to those shown in Figure 5. Again threat intensity is color coded and targets are indicated by black and red dots, where red indicates a successfully prosecuted target.

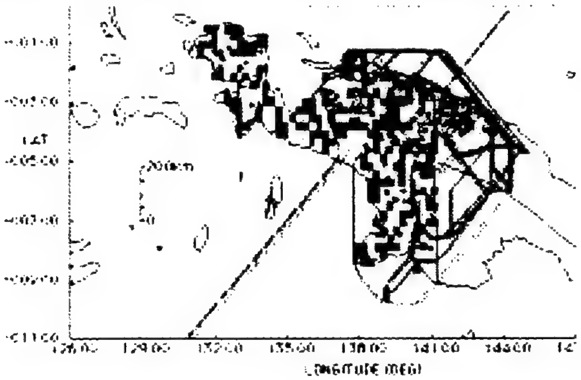


Figure 6 Simulation of Planned Strike Missions.

5.1.1 Baseline Cases 4 hr vs. 24 hr

Our baseline experiments investigated the performance differences between the 4 hour controller loop-closure cycle and 24 hour cycle for both a 313 target scenario and a 910 target scenario.

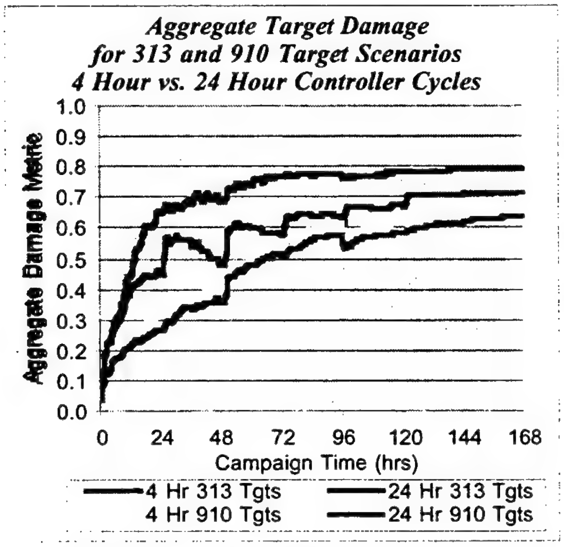


Figure 7 Baseline313 and 910 Target Scenarios for 4 Hr and 24 Hr Loop Closure Intervals

Figure 7 shows that the four hour cycle provides better performance in terms of the aggregate damage achieved over time for both scenarios, with the greatest difference being for the 313 target case for the 7 day campaign time. The high frequency ripple in the results is a combination of new target disclosures from ISR resources at 4-hour intervals, reducing the (normalized) aggregate damage metric and the damage visited upon targets by successive waves of strike packages. The automated controller plans the dispatch and arrival on target with specified routes through assembly points, tanker locations, and waypoints for threat avoidance. The risk along each route depends on the escort package and some missions are deferred depending on the settings for risk aversion. The rate at which packages can be dispatched depends on aircraft turnaround times as well as flight times.

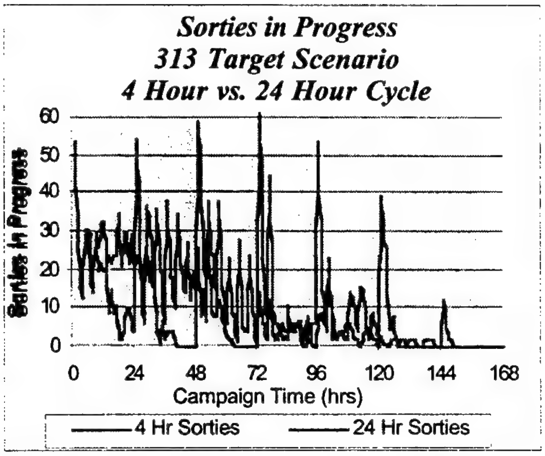


Figure 8 Sorties Generated by 4 Hr and 24 Hr Cycles for 313 Target Scenario

Note in Figure 7 that for the 24 hour planning cycle, successive waves of package sorties (see Figure 8) exhaust the targets known at the beginning of the day and add a 24-hour structure to the performance metric. The 4 hour controller cycle exhibits a more level use of strike package resources over time and results in striking nearly all of the known targets that are feasible by the end of the fifth day. The 910 target scenario stresses the ability of the assumed deployment to generate sorties.

Since the attrition of strike packages is simulated probabilistically, we have made a series of Monte Carlo experimental runs (Figure 9) to determine the dispersion of results. Statistical analysis has shown that the standard deviation of the aggregate damage metric varies from its mean value over time with values between 0.02 and 0.05.

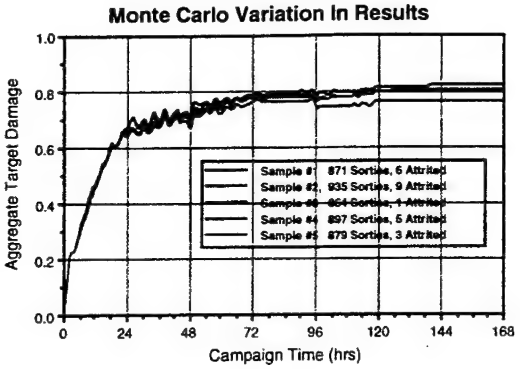


Figure 9: Dispersion of Aggregate Damage for Monte Carlo Trials of 313 Target Scenario

5.1.2 Impact of Problem Decomposition

A series of experiments were run wherein strike plans were generated using both decomposed and undecomposed solutions. Undecomposed solutions were developed for both the 313 and 910 target scenarios. Computation times were reduced for the decomposed solutions compared to the undecomposed solutions by roughly a factor of 8. Indeed, decreased computation time alone for the decomposed solution would not be of benefit if the plans developed were substantially inferior to those generated by the undecomposed solution. Figure 10 indicates that there was little loss in performance for the decomposed cases compared to the undecomposed for both scenarios.

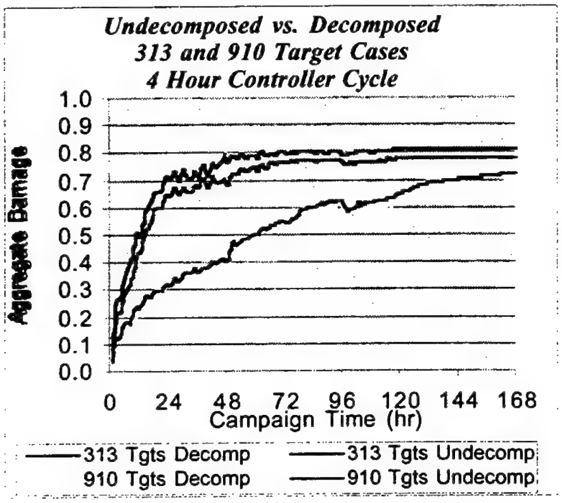


Figure 10 Performance of Undecomposed vs. Decomposed Problem Solutions

This as well as all other experiments reported here has been run with the heuristic Level 1 and Level 2 planners. We expect to see much greater relative improvements in computation time with decomposition when the optimal planners are used since the computation time for the optimal algorithms increases exponentially with problem size as opposed to the polynomial increase for the heuristic solutions.

5.1.3 Sensitivity to Modeling Errors: 4 vs. 24 hr

Two sets of experiments for the 313 target scenario were developed to determine the ability of the closed loop controller to respond to errors in its model of the operational environment. In the first set, the controller underestimated the intensity of the air defense threat by a factor of two. Compared to the baseline case, this perturbation showed a small increase in attrition and slightly better aggregate damage performance. This apparent anomaly (improved performance due to a modeling error) occurs because the strike packages are, in effect, less risk averse in comparison to the baseline case due to the threat intensity mismodeling and achieve a higher damage performance at the expense of higher attrition. The four hour cycle exhibited significantly better performance than did the twenty four hour cycle.

For the second mismodeling experiment the controller overestimated the effectiveness of its weapons ability to damage targets (e.g., the targets were harder than expected). The controller operating at the 4 hour cycle is able to respond more quickly to it's more frequent, higher rate battle damage reports and thereby is able to re-attack high value targets.

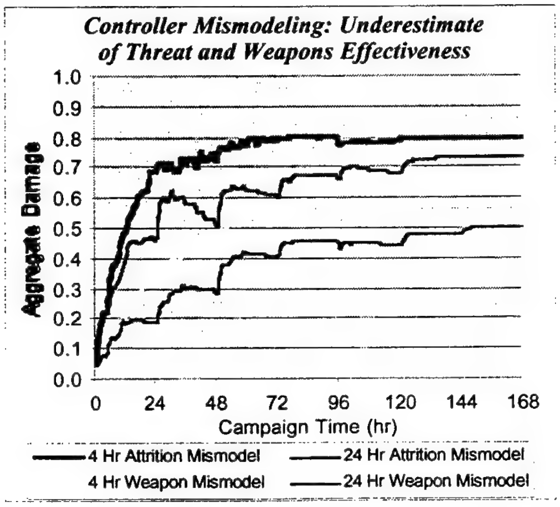


Figure 11: Modeling Errors: 4 Hour vs. 24 Hr Cycle

In both cases (4 and 24 hr cycles), the performance is significantly degraded in comparison to the baseline experiments. We expect that higher loop closure rates would be required to respond quickly enough to impact the overall performance which is influenced by the controller's ability to prosecute time sensitive targets (see the next experiment).

5.1.4 Time Sensitive Targets

Time sensitive targets arise because of fleeting opportunities to acquire and deliver weapons on relocatable targets and also because of fleeting opportunities to negate activities that may be occurring at fixed target locations. Targets with intrinsic dynamics such as time perishability of the probability of acquisition and time reconstitution for repair of damage are modeled by first-order, exponential models with a single time constant.

The scenarios under study contained approximately 10% of all targets in the "time-sensitive" category, including some airfields with a 12 hour time constant and long-range artillery and maneuver units with 4 hour time constants. The controller would attempt to deliver weapons on these targets for up to 5 time constants between the time of disclosure and the time on target. Missions to address these targets could involve missions for uncommitted aircraft, aircraft that were in preparation for another mission but not yet dispatched, or enroute diversions if the priority was high enough, the right weapons remained on wing and the rerouted risk was acceptable.

To more thoroughly understand the effects of controller loop closure rate, additional cases were defined *without any* time sensitive targets and also *with only* time sensitive targets, with results shown in Figure 12.

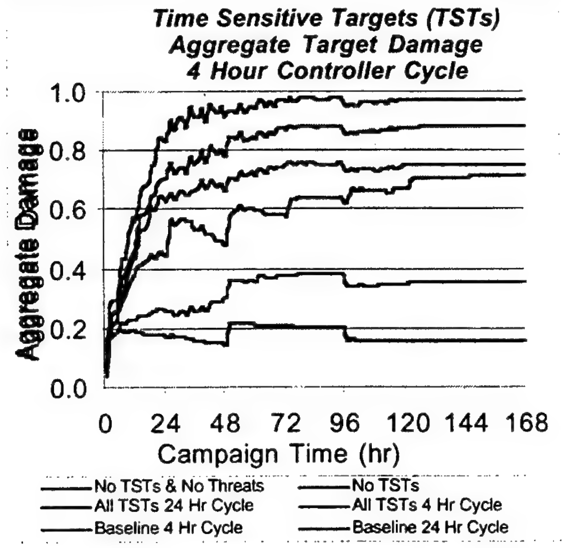


Figure 12: Time Sensitive Target Cases

Without targets disappearing and without threats, virtually all of the targets are accessible to prosecution as shown in the upper curve. With the normal threat laydown and the routing for risk mitigation that is required, approximately 10% of the target set is not accessible to prosecution, even without time perishability, for the specified tanker locations, aircraft ranges, and weaponeering requirements. The middle curves show the nominal performance for 4 vs. 24 hour loop cycles, with the former performing significantly better overall, and especially in the category of time sensitive targets. The limit of this improvement is shown in the lowest curves where a factor of two is scene for the difference between 4 and 24 hour cycles for the case of all time sensitive targets.

5.1.5 Summary Results: 4 hr vs. 24 hr

A significant and major result that we have observed through our experimentation is that the ability to close the loop at higher rates ***shortens the total time required to achieve campaign objectives***. To quantify this effect, we have performed a least-squares fit through the aggregate target damage vs. time curve and extracted the difference in time to achieve different levels of aggregate target damage using the 4 hour vs. the 24 hour loop closure rate. The results are shown in Figure 13 below. The benefit for the baseline case ranges from 12 to 48 hours for typical ranges of target damage objectives, a very substantial gain. This benefit accrued in cases of modeling error as well as for dynamic scenario changes. For the case of the weaponeering error, the benefit could be even larger. For the case of the capacitated 910 target scenario, the benefits ranged from 8 to 24 hours.

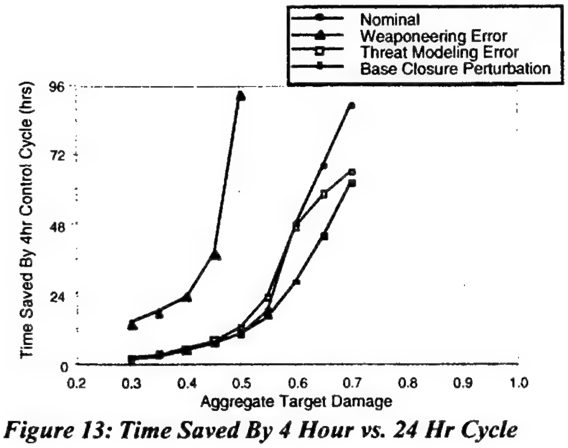


Figure 13: Time Saved By 4 Hour vs. 24 Hr Cycle

6 Conclusions

There has been a long-standing advocacy by those in the command and control community for closing the command and control feedback loop at ever shorter intervals. This advocacy relates to a *rule of thumb* in control system design: that is, to achieve good system performance and maintain robustness, the total time lag for each control cycle attributable to (a) measuring and conditioning feedback signals, (b) information transfer and (c) control law computation should be one-fifth to one-tenth the time constant of the fastest dominant mode of the plant to be controlled. Applying that rule of thumb to the control of military air operations, our goal should be command and control cycle times that are five to ten times shorter than those of our adversaries. Although the differences between the nature of the plant to be controlled by a traditional closed-loop controller and that to be controlled by a military command and control system are significant, there are obvious advantages in an ability to plan, execute and replan many times faster than one's enemy.

The work reported in this paper is one of the first instances where the benefits of higher rate loop closure have been quantified for a complex enterprise command and control application such as coordinating air attack operations spanning the air operations enterprise from JFACC level to the strike package level. Our experimental results show that the benefits are substantial and that they accrue even in the face of the types of model discrepancies that are to be expected in such applications.

We should note that the results reported here assume perfect state estimation and feedback for own forces as well as for battle damage assessment. Future experiments will investigate the extent of performance degradation that would be experienced in the face of significant latency or noise on the feedback signal.

Further planned work includes the completion of our development of optimal solutions to the Level 1 and Level 2 problems of Target Partitioning and Mission Generation and comparison of those solutions to the experimental results for the heuristic solutions discussed here. In addition, we will continue to work on the auxiliary components of our closed-loop controller architecture including: plan monitoring, diagnosis and interlevel coordination.

7 References

1 Hall, William, D. and Adams, Milton B., "Closed-loop, Hierarchical Control of Military Air Operations," *Proceedings of Symposium on Advances in Enterprise Control,* San Diego California, November 14-15, 1999.

2 Singh, M. H. and A. Titli, Systems: Decomposition, Optimization and Control, Pergamon Press, Oxford, 1978.

3 Tamura, H. and T. Yoshikawa, eds., *Large-Scale Systems Control and Decision-Making*, Marcel Dekker, New York, New York, 1990.

4 Lasdon, L. S., *Optimization Theory for Large Systems*, Macmillan, New York, New York, 1970.

5 Mahmoud, M.S, Hassan, M.F. and Darwish, M, G., *Large-Scale Control Systems: Theories and Techniques*, Marcel Dekker, New York, 1985.

6 *Proceedings of Symposium on Advances in Enterprise Control,* San Diego California, November 14-15, 1999

7 Armacost, A. P., *Composite Variable Formulations for Network Design.* Ph.D. dissertation, Massachusetts Institute of Technology, 2000.

State-Based Abstractions in Message Flow Models

Paul Hubbard
Logica Carnegie Group
5 PPG Place, Pittsburgh PA 15222
hubbardp@logica.com

Abstract

A preliminary investigation of state-based hierarchies for message flow systems is presented. Two abstraction systems are considered: (i) a stepwise refinement of Petri Nets which appeared in [15, 14] in which single transitions are refined with so-called *well behaved* Petri Nets and (ii) a state-based aggregation approach for supervisory automata which appeared in [2, 4] in which partitions of the state space are used to form high-level automata models. The two techniques are contrasted and compared in an example and theoretical connections between the two theories are conjectured.

Keywords: discrete event systems, Petri Nets, supervisory control, hierarchy, aggregation.

1 Introduction - Complexity and Hierarchies

Mechanisms for creating consistant abstractions are vital for managing the organisational and constitutional complexity ([12]) that arise in enterprise control systems whether they be military enterprises, manufacturing plants or communications networks. These aspects of complexity play a more important role than that of computational complexity in early stages of design or analysis. Most often it is not the length of time of calculations that is important at this stage, but rather the need for an organised approach to combining subsystems that have extensive interactions.

A natural response to the issue of organisational complexity is hierarchical organisation. It has been argued that hierarchies are universally present in natural and synthetic complex systems [13, 5]. Characteristics of multi-level hierarchical structures, their vertical arrangement, and subsystem prioritisation have been defined in various contexts [8, 16].

Two approaches to the formation of hierarchies in Discrete Event Systems (DES) are discussed in the following sections. The first is in the context of Petri Net (PN) models which have been proposed for a wide variety or enterprise control applications [1, 9]. An accepted difficulty with the use of the PN model for systems of high constitutional complexity is that the number of places and transitions becomes too large for the model to be of use as a design tool. A method for refining and abstracting PN representations was proposed in [15] and is briefly summarized in Section 3. An analysis of the existence of supervisory policies (i.e. policies that disable transitions based on the transition history) that enforce liveness at different various levels of abstractions for these refinements was presented in [14].

The second approach is in the context of the supervisory control framework for modelling DES. This is an untimed logical model that is expressed in terms of the observation and inhibition of events. Within this framework, system behaviours are described by languages (i.e. sets of strings of events) and the theory seeks to determine which behaviours can be achieved via a supervisor that may inhibit a subset of the system's events (see [11] and references therein, in particular [10]). A hierarchical theory for supervisory control based on state-aggregation appeared in [3, 4]. In this theory, conditions are determined on state partitions which ensure that the control of transitions between blocks in a high-level (i.e. aggregated) model, combined with local state-dependent controls is effective in the sense of achieving specifications given either for the high-level model or for the low-level system.

It should be noted that trace-based (rather than state-based) approaches to hierarchical control of DESs have also appeared in the literature ([17, 18]). Other theoretical research at the boundary of Petri Net models and supervisory control exists in the context of vector discrete-event systems ([6, 7]).

The main contributions of the present work are

A to present the formulations in [3] and [15] in the context of models for message flow in enterprise control systems, and

B to compare and contrast the consistency conditions developed in the two formulations in order to postulate a theoretical connection.

2 Motivating Example - Multi-Unit Message Processing Center

Consider first the abstract process "Process 1" illustrated at the bottom of Figure 2. This process represents, at a very coarse level of abstraction, the need for sequencing in an arbitrary enterprise process, e.g. targetting in a military domain or tracking sales in a commercial domain.

The hashed lines in Figure 2 illustrate a refinement of the decision-making unit DMU2 with "subprocess 2". The re-entrant flow in subprocess2 reflects an abstract "subordinate-supervisor" relationship as messages processed by DMU4 and DMU5 must be tested by the supervisor DMU6 and may be accepted (and hence given authority to proceed) or rejected, the latter resulting in another pass through the first two units. To emphasize the fact that the level of refinement can be further increased, a more complex "Subprocess 3" is also illustrated in the figure.

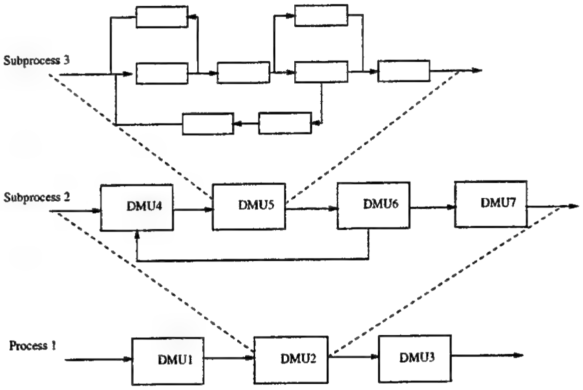


Figure 1: A sequential process and a subprocesses exhibiting a "subordinate-superivosr" relationship

There are many issues regarding consistency that arise in this refinement process. How can one can be assured that a refinement is consistent with the original model, i.e. is such that the original model, with refinements masked, continues to evolve subject to the formal protocol that was originally posited. As a specific case of this, it is expected that messages processed by DMU2 should be forwarded to DMU3 and additional messages should not be created artificially by the internal workings of DMU2. Note that in subprocess3 this situation occurs. DMU5 us able to create messages artificially by circulating current messages.

The motivation in this paper is to use the theories suggested in [15] and [3] to provide a methodology for sequential refinement that ensures consistency between real and abstract models with respect to such measures as maintanance of messages.

3 Stepwise Refinement of Petri Net Models [15]

In this section, the intention is to supply a subset of the analysis in [15] required for comparisons with the following section hence only the essential definitions and interpretations are given. The reader is referred to [15] for the formal definitions omitted here and to [9] for more detail on Petri Nets.

Definition 3.1 Petri Net

A Petri Net (PN) is a five-tuple $N = (P, T, \Phi, M_0)$, where
P is a finite set of *places*,
T is a finite set of *transitions*,
$\Phi \subset (P \times T) \bigcup (T \times P)$ is a finite set of arcs, and
$M_0 : P \longrightarrow \mathcal{N}^+$ is an initial marking (where $\mathcal{N}^+$ are the non-negative natural numbers.

The marking, or "state", of the PN at time i is a map $M_i : P \longrightarrow \mathcal{N}^+$, capturing the number of 'tokens' at each place in P. Informally, the PN evolves by firing any *enabled* transition t in T and updates the marking by moving tokens between places. The set of arcs is used to identifty from which places a given transition draws tokens and to which it delivers tokens. The standard graphical representation of a PN uses circles for places and rectangles for transitions and connects these with directed edges representing the arcs.

The key definitions of relevence in [15] are now informally paraphrased:

- A transition $t \in T$ is termed *k-enabled* if there exists a marking reachable from M_0 and from which t can be fired immediately k consecutive times.
- A transition t is *live* if it can eventually be enabled from all reachable markings,
- Given two transitions, t_{in} and t_{out}, in a PN N, define a new Petri Net $B(N, t_{in}, t_{out}, k)$ as illustrated in the following figure, i.e. connect the place p_0 to t_{in} and t_{out} and place k tokens in p_0.

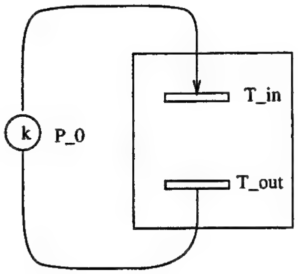


Figure 2: Forming $B(N, t_{in}, t_{out}, k)$ from N

- A Petri Net N is *k-well behaved* if

[a] t_{in} is live in $B(N, t_{in}, t_{out}, k)$, i.e. t_{in} never gets blocked

[b] t_{in} can only have fired up to k times more than t_{out} in any given evolution of $B(N, t_{in}, t_{out}, k)$.

The fundamental result from [15] that should be stressed is that if a transition in a PN N that is not $(k+1)$-enabled is replaced with a *k-well behaved* PN (yielding a refined PN N'), the following will be true: (i) any firing sequence in the original PN N can be simulated by some firing sequence in N' and, conversely, (ii) any firing sequence in PN N' is a simulation of some firing sequence in N.

4 State Aggregation Based Hierarchical Supervisory Control [3]

The theory from [3] will be demonstrated with an example (taken from [3, 4]) that bares a strong resemblence to the process illustrated in Figure 2.

Consider the layout for a transfer line with re-entrant flow shown in Figure 3. The automaton model for the layout can be found by taking the synchronous product of the individual machine and buffer models from figure 4. In Figure 5, a portion of the low-level system is displayed (the full model has 129 states). The state $\langle IIII000\rangle$ is identified as the initial state as it is the "empty" state. The evelution of the state of the system can be traced on this graph, for instance, as an event "1" occurs, a piece enters the first machine and a corresponding transition occurs in Figure 5. A natural partition based on the number of active pieces is also displayed in Figure 5 leading to an abstract automaton representation in Figure 6.

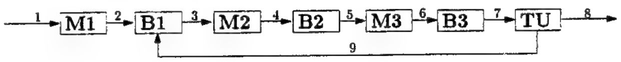


Figure 3: A material transfer line with re-entrant flow

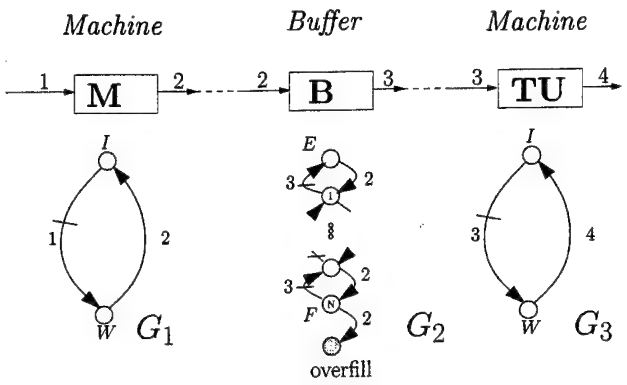


Figure 4: The machine, buffer and testing unit models

The main result of [3] states that the control, at the abstract level, which restricts the reachable set to the shaded blocks in Figure 6 can be realized in the low-level system.

Notice, for instance, that prior to supervision the set of possible events that can occur at x_i in Figure 5 includes the event 1. The low-level control inhibits 1 at x_i to prevent flow to the block "X_5: 3 Pieces" as U_4^5 is inhibited at the abstract level.

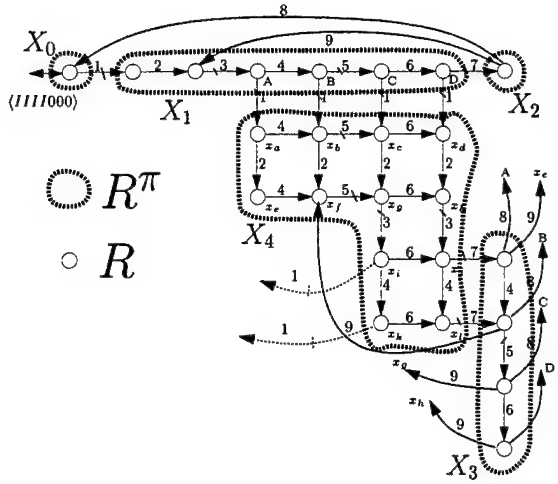


Figure 5: A portion of the state space for the plant in Figure refplant a state partition

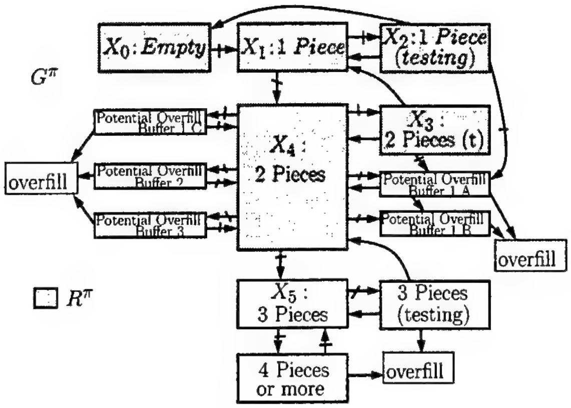

X_0: *Empty*
X_1:1 *Piece*
X_2:1 *Piece (testing)*
G^π
Potential Overfill Buffer 1 C
Potential Overfill Buffer 2
Potential Overfill Buffer 3
overfill
X_4 : 2 Pieces
X_3 : 2 Pieces (t)
Potential Overfill Buffer 1 A
Potential Overfill Buffer 1 B
overfill
□ R^π
X_5 : 3 Pieces
3 Pieces (testing)
4 Pieces or more
overfill

Figure 6: An abstract representation and "good" state-set for the plant in Figure 3

5 A summary of the application of the two theories to the process example

To aid in the application of the work in [15] to the example process described in Figure 2, a PN version of the process is supplied in Figure 5. Note that DMU2 in this figure is a k-well behaved PN for all k. Hence an abstraction that is consistent with respect to liveness and the representation of firing sequences can be created by replacing DMU2 with a single transition. It is exactly this kind of technique that is required, at the design stage, as a tool for the analysis of these complex systems.

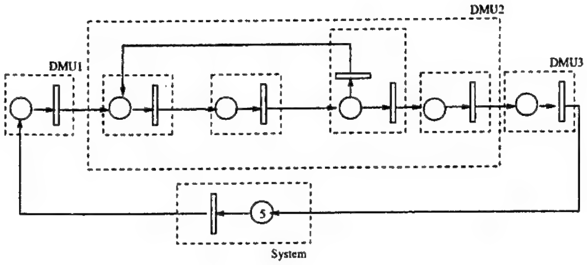


Figure 7: A Petri Net model of the message process in Figure 2

The application of the theory in [3] to the process in Figure 2 would appear much more straightfoward in the light of the example provided in the previous section. The goal is to enable the control of message flow at a high level of abstraction in order to contain complexity explosion.

Finally, as a theoretical connection between the two theories, it is suggested that the *well-behaved PN* description in [15] can be harnessed to create the partitions for the states (in the PN setting, these are the markings) suggested in [3]. A potential method to accomplish this is to group markings that are equal (in terms of number of tokens at each place) within the *well-behaved PN*. It is conjectured that this will give what is termed *In-Block-Controllable* partitions in [3].

6 Acknowledgements

The author gratefully acknowledges the time provided by Logica Carnegie Group for the preparation of this submission, and the many constructive

discussions held with co-workers at that location.

References

[1] T. Agerwala. Putting petri nets to work. *Compute*, 12:85–94, 1979.

[2] P. Hubbard and P.E. Caines. A state aggregation approach to hirarchical supervisory control with applications to a transfer line example. In *Proc. of the WODES98: Workshop on Discrete Event Systems*, Cagliari, Italy, August 1998. IEE.

[3] P. Hubbard and P.E. Caines. Trace-*dc* hierarchical supervisory control with applications to transfer-lines. In *Proceedings of the 37th IEEE Conference on Decision and Control*, pages 3293–98, Tampa, FL, 1998.

[4] P. Hubbard and P.E. Caines. Dynamical consistency in hierarchical supervisory control. *Submitted to IEEE transactions on Automatic Control*, 1999.

[5] A. Koestler. *The Ghost in the Machine.* The Macmillan Company, 1967.

[6] Y. Li and W.M. Wonham. Control of vector discrete-event systems i-the base model. *IEEE Transactions on Automatic Control*, 38(8):1214–1227, August 1993.

[7] Y. Li and W.M. Wonham. Control of vector discrete-event systems ii-controller synthesis. *IEEE Transactions on Automatic Control*, 39(3):512–531, April 1994.

[8] M.D Mesarovic, D. Macko, and Y. Takahara. *Theory of Hierarchical, Multilevel, Systems.* Academic Press, 1970.

[9] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77:541–580, April 1989.

[10] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event systems. *SIAM J. Control and Optimization*, 25(1):206–230, 1987.

[11] P.J. Ramadge and W.M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, January 1989.

[12] N. Rescher. *Complexity, A philosophical overview.* Transaction Publishers, 1998.

[13] Herbert A. Simon. *The Sciences of the Artificial.* The MIT Press, 1996.

[14] R. S. Sreenivas. On supervisory policies that enforce liveness in a class of completely controlled petri nets obtained via refinement. *IEEE Transactions on Automatic Control*, 44(1):173–177, January 1999.

[15] I. Suzuki and T. Murata. A method for stepwise refinement and abstraction of petri nets. *Journal of Computer and System Sciences*, 27:51–76, 1983.

[16] J. Talavage and B. Elliott. *Toward a theory of hierarchical coordination and conflict*, volume Disaggregation, problems in manufacturing and service organizations. Martinus Nijhoff Pub., 1979.

[17] K.C. Wong and W.M. Wonham. Hierarchical control of discrete-event systems. *Discrete Event Dynamical Systems*, 6:241–273, 1996.

[18] H. Zhong and W.M Wonham. On the consistency of hierarchical supervision in discrete-event systems. *IEEE Transactions on Automatic Control*, 35(10):1125–1134, 1990.

Adapting the Linguistic Geometry-Abstract Board Games Approach to the Air Operations[1]

Boris Stilman

CS & E Dept.
University of Colorado at Denver
Campus Box 109, Denver, CO
80217-3364
bstilman@cse.cudenver.edu
tel: (303) 556-3614
fax: (303) 556-8369

Vladimir Yakhnis

Rockwell Science Center, M/S
A30
1049 Camino Dos Rios
Thousand Oaks, CA 91360
vryakhni@rsc.rockwell.com
tel: (805) 373-4856
fax: (805) 373-4862

Abstract

Linguistic Geometry (LG) is an approach to finding "good enough" strategies providing solutions for various kinds of abstract board games. The LG approach employs several mathematical tools the most important of which are Zones, trajectories, and strike maps. We provide here a new formal definition of ABG and new LG algorithms for strike maps, shortest trajectory bundles, strike trajectory bundles, and Zone bundles. We also discuss these formalisms in the light of their adaptation to the Air and military operation domains. We compare LG with other game methodologies.

1 Introduction

1.1 System Control via Abstract Board Games (ABG) and Strategies

This paper continues the theme started in [12]. To solve the problem of discrete system control, we view the state transition process of a system as a game between several abstract players. This game need not be strictly adversarial and may involve cooperation as well as contest. (Note that since the players discussed here are abstract entities, we may sometimes refer to a single such player as "it".) Within the game model, each abstract player may have its own goals and it is usually able to exercise at least a partial influence over the state transitions.

Of course, the gaming model is not the only way to describe discrete system behavior, e.g., finite state machines could be employed to hide the game metaphor. Without the game metaphor, the players are sometimes called agents and the system where such agents are present is called a multi-agent system. What makes the games particular convenient is the notion of a strategy for a player, especially a strategy with restricted memory or a state strategy [13,15,16]. Informally, a strategy is a function such that, given a previous history of the game, outputs one or more legal actions (also called "moves" in game terminology) for the player.

A state-strategy for a player ω_i is an input-output automaton that accepts the moves of all the players other than ω_i as input, and that outputs the moves for ω_i. Input is used to memorize relevant information about the play. Output is used to guide the behavior of ω_i during the play. A state-strategy for alternating two player games is illustrated on Figure 1.

Usually, we would like to supply some of the players with the strategies helping them to achieve their goals. If there is only one such player, it may be designated as the Friend. If there are only two abstract players, we may call the other player by various means, e.g., the Second Player, the Opposing Player, or the Adversary, depending on how close its goals would be to the diametrical opposition to the Friend's goals.

Further, we will limit ourselves to abstract board games (ABG). Within the ABG area, the game (or system) state is modeled by placement of various objects (called pieces

[1] This research is supported in part by a DARPA contract N66001-99-C-8509.

or local mobile agents) on an abstract game Board. The position of pieces on the Board may have a profound impact on their movements and/or actions. E.g., for a given position of a piece, some of the locations on the Board may be reachable in a certain number of steps, whereas others may not be reachable at all. The state transitions of the game are defined via movements of the pieces on the board, addition of new pieces, or elimination of some pieces from the board, see Figure 7. ABG were defined in [15,11] by expanding on graph-games defined in [13,16].

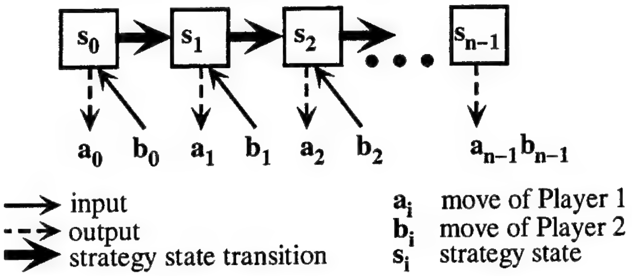


Figure 1. Application of a state-strategy in an alternating two player game

It is important to stress that an ABG is not necessary a zero sum game, e.g., if there are only two players, their goals need not be diametrically opposing. Thus, it is possible for the players to both win or to both lose. Sometimes a zero sum game emerges when we try to model a real world situation when we just don't know the true purposes of the opposition. In that case, we would assume that the Adversary's goal is to guide the controlled process to a point where the Friend would fail.

In general, we divide the ABG into the following three classes:

- An *Alternating Serial* (AS) ABG is an alternating game where only one piece at a time can move or act. The combined move also includes all those pieces of any player that are moved or destroyed as an immediate result of the actions of the aforementioned single piece. The players take alternate turns. Examples are Chess, Checkers, etc.;
- An *Alternating Concurrent* (AC) ABG is an alternating game where all, some, or none of the pieces of one of the sides can move or act simultaneously. The combined move also includes all those pieces of any player that are moved or destroyed as an immediate result of the actions of the aforementioned pieces. The players take alternate turns. There are few examples of such games;
- A *Totally Concurrent* (TC) ABG is a game where all, some, or none of the elements of both sides can move simultaneously. As before, the combined move also includes all those pieces of any player that are moved or destroyed as an immediate result of the actions of the aforementioned pieces. An example is a real world battlefield.

In practice, the game description must provide the following:

– a description of the game Board and a mapping between the board locations and the actual conflict region, thus dividing the conflict region into cells;

– rules, defining the legal motions and other actions for each piece, as well as its interactions with other pieces, including the enemy pieces;

– initial placement of the pieces on the Board;

– various constraints on the game actions, such as "no fly zones";

– various conditions determining the game outcome, such as the stopping condition (defining when to stop playing), the winning condition (defining when a given player wins) and others, such as mission abort conditions.

In general, the ABG approach is applied as follows. First, the problem is defined in ABG terms, i.e., the players, the Board, the pieces, the game rules, etc., are identified. Then some methods are utilized to generate strategies that would guide the behavior of the designated players (i.e., the Friend and its allies) so that their goals would be fulfilled. We intend to utilize a unique approach for generation of state strategies called Linguistic Geometry (LG), see [7,8,9,10,11,15].

Linguistic Geometry (LG) is a theory for solving classes of abstract board games (ABG), which include problems of air/space combat, robotic manufacturing, Internet Cyberwar, etc. This report includes an introduction to ABG, as well as major definitions of two formal languages, Trajectories and Zones, which allow us to decompose an ABG into the hierarchy of subsystems. It also includes extensions to ABG, Trajectories, and Zones pertaining to the JFACC specifics.

The LG approach is applicable to a wide class of multi-agent systems, containing both sequential and concurrent systems. Although the LG approach can represent each of the AS, AC, or TC classes of games (see [13]), in this report we will concentrate on the TC class. There is another system characterization of interest to the domain of military operations – that of centrally controlled vs. autonomous behavior of pieces. In contrast to concurrency, the degree of decentralization is not well represented on the level of ABG definition. Rather, it may be represented on the level of strategy implementation. With respect to this, the LG approach can represent the paradigms either of centralized control or of various levels of decentralization. It may be done as follows. To represent a player, say ω, centrally controlling all of its pieces, we just assign the LG strategy to ω, making the pieces to obey all of its dictates. To represent various levels of decentralization of ω's pieces we may do the following:

- introduce additional players called the sub-players of ω;
- constrain them to be ω's allies by subordinating their goals to that of ω;
- assign subsets of the pieces of ω to sub-players;
- let each sub-player control its pieces via an LG strategy.

Thus, collectively, the sub-players will strive to fulfil the goals of ω but they will not be under central control by ω. The level of decentralization could go up to assigning a sub-player to a single piece. These various levels of decentralization correspond to behaviors of battlefield units that are sometimes centrally controlled and sometimes autonomous.

1.2 Global Operations Command and Control (C^2)

Linguistic Geometry (LG) is an approach to the construction of mathematical models for knowledge representation and reasoning about large-scale multi-agent systems [11]. A number of such systems, including air/space combat, robotic manufacturing, Internet Cyberwar, etc., can be modeled as abstract board games (ABG). These are multi-player games, whose moves can be represented by means of moving abstract pieces over locations on an abstract board. The dimensions of the board (2D, *n*D, and even non-linear space), its shape and size, the mobility of pieces, the player turns of making moves (including concurrent moves) – all can be tailored to model a variety of multi-agent systems. The purpose of LG is to provide strategies to guide the participants of a game to reach their goals. Traditionally, finding such strategies required searches in giant game trees. Such searches are often beyond the capabilities of modern and even conceivable future computers.

LG dramatically reduces the size of the search trees, thus making the problems computationally tractable. To achieve that, LG provides a formalization and abstraction of search heuristics of advanced experts in the form of the game strategies. Essentially, these heuristics replace the search by the construction of such strategies. The formalized expert strategies yield efficient algorithms for problem settings whose dimensions may be significantly greater than the ones for which the experts developed their strategies. Moreover, these formal strategies proved to be able to solve problems for different problem domains far beyond the areas envisioned by the experts. These strategies are not intended to provide solutions that are always optimal, but they are intended to provide "good enough" solutions. Although for some classes of problems, these formalized expert strategies yield provably optimal solutions, for the rest of the problems the LG strategies are nearly optimal. To formalize the heuristics, LG employs the theory of formal languages (i.e., formal linguistics), as well as certain geometric structures over the abstract board. Since both the linguistics and the geometry were involved, this approach was named Linguistic Geometry.

LG-ABG can be utilized to model and assist the military operations at various levels of resolution, see Figure 15. At the top (strategic) level, the lowest resolution model controls the global planet-size operations, as well as the largest possible teams of military mobile units. The full spectrum of mobility of those teams is employed. In the LG-ABG terms, the LG "operational domain" (an abstract Board) would be a low-resolution grid that embraces oceans, land, air, near-planet space, etc. The agents would be friendly and opposing teams of submarines, ships, mobile military units on land, air force units, and space assault vehicles. The LG "reachability relations" would permit us to reflect the mobility and military strength of teams ranging from under-water sailing of submarines to orbit-changing maneuvers of assault satellites. A hierarchy of higher resolution models control smaller teams and separate vehicles, while focusing on smaller operational districts like marine-land, air-land, or classes of space orbits.

A mission preparation with LG models could be conducted by running multiple experiments, so that the mission commander ("the man in the loop") may select the best Initial State, i.e., the best initial configuration of all the friendly agents to be involved in the operation. After the Initial State is selected, LG application would generate an initial strategy for the mission. After the actual engagement starts, the mission execution control would be conducted in real time as follows. In the beginning, the initial LG strategy would be utilized to provide advice for the commander. As the mission progresses, the LG strategy would be re-tuned by taking into account the actual advancement of agents, actual losses/gains, and changes of mobility, as well as the actual enemy actions. Similar mission preparation and real-time control of mission execution may be conducted on a smaller scale by each team and each military unit reflecting their autonomy or subordination (see discussion on decentralization in section 1.1).

A variety of computers at the battlefield may be linked over the network. This would permit us to coordinate several LG battlefield advisors on various levels of abstraction, see Figure 15. A global mission may require extensive coordinated actions including underwater maneuvering, space satellite hunting, etc. A number of LG strategies may take advantage in strategic patterns developed beforehand by the military experts (either LG-assisted or not) and stored in a database. These retrieved strategies and patterns would allow us to utilize the historical experts knowledge by identifying strategies leading to familiar patterns of successful operations and by avoiding strategies leading to known failures. If they would not available, the LG would provide advice based on the current situation only.

On the other hand, the new original strategies generated via commanders assisted by LG in the course of a global mission may be memorized in a database at the end of the mission. These strategies would carry the significant knowledge and skills of the domain experts, ranging from Lieutenants to Chiefs of Staff. The knowledge of such experts would be stripped of unimportant details, formalized and included in the general LG framework. Moreover, the enriched LG strategies would also allow us to transfer this knowledge to a variety of different problem domains.

2 Representation of Continuous JFACC-Related Problems as Abstract Board Games (ABG)

2.1 The Pieces as Mega-Groups of Aircraft

Instead of dealing with individual aircraft, at various levels of the hierarchy of game boards the pieces represent "mega-groups" of aircraft representing the air battle units appropriate for the level. Figure 16 illustrates the possible modeling entities in the hierarchy and their relationships. At the lowest level in the hierarchy are individual components. These components represent the smallest physical entities *of interest* in military air operations that can be controlled and coordinated to perform certain functionality. In general, the categorization of an individual component is related to the actual unit of operational interests during the air operations and can sometimes vary in terms of granularity. Hence, depending on the mission scale and characteristics, a component can be either a single aircraft or a fighter squadron (where, in this case, individual planes will be treated as resources/capability of this squadron component).

The entities at the next level in the hierarchy are groups. A group consists of several individual components that are coordinated to achieve a certain common objective/task. Based on the granularity of the components, examples of groups can be either a bomber group assigned to attack specific ground targets or a fighter wing deployed to defend a designated airspace. Individual components within a group can be of different types so long as they can be tasked to achieve a common objective (e.g., a composite wing).

A group of groups is a "mega-group" and it represents the top layer of the hierarchy. The interaction/coordination among groups can be treated as the behavior of a mega-group where each group is viewed as a composite "component" of the mega-group. A mega-group can be formed either based on the commanding hierarchy or when specific interaction/ coordination among certain groups (or mega-groups) needs to be addressed. We denote the mega-group that contains all the entities assigned to the mission as the top mega-group.

2.2 Hexagonal Prism Cells (Hexes)

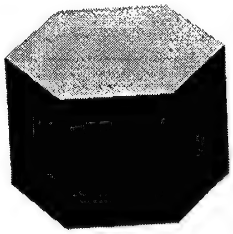


Figure 2. 3D hex cell

We would like to cover the domain of the Battle Theater by fixed size cells in the shape of hexagonal prisms, see Figure 2. We will employ the cell numbering system (m,n,k) where m,n is the numbering within the horizontal layer, with m as the column number and n as the row number, and k is the layer number. The horizontal numbering is illustrated on Figure 3.

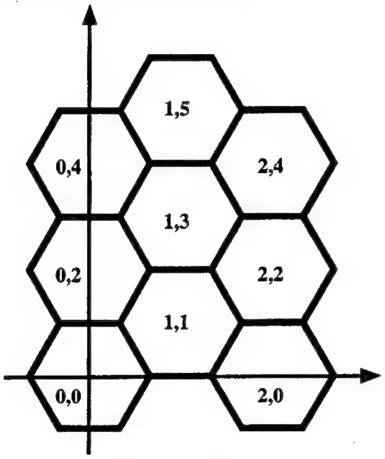


Figure 3. A Horizontal Projection of the Hex Grid

2.3 3D Layers

For the Air operations, we propose to break the continuous 3D space into 4 layers where the air operations take place:

- The ground layer where some of the targets are located;
- The low altitude layer, where the attacking aircraft may hide behind the obstacles while moving to the target area (about 1500-3000 ft above ground);

- The middle layer, where most attacks take place (12000-15000 ft above ground);
- The high altitude layer, where the aircraft can move with the least expenditure of fuel (about 40000 ft above sea level).

Although the Game Board will take into account the space between the layers, only the limited activities would be undertaken between the layers and the trajectories there would assume fixed rates for ascent and descent. Note that the thickness of the layers is different as measured by the cell heights.

2.4 Reachability Relations

2.4.1 Motions Reachability

In variance with the standard LG Theory, [11], we will consider pieces with velocity vectors that may change from one move to another. To simplify matters, we assume that each piece, i.e., a group of aircraft, may assume only a few possible speeds, e.g., the cruising speed and the maximum speed. Each would be determined as the lowest respective speed (i.e., cruising or maximal) of the lower level units in the group (e.g., individual aircraft). We will also assume (for now) that the pieces accelerate only along a straight line. This will impose certain constraints on the reachability relation.

In order to determine the reachability relation, we would want to be sure that the craft would indeed be able to reach from point A to point B in one move. Now, since the aircraft in question moves at high speed, certain maneuvers, like instant 180 degree turns, are impossible. To simplify the computations, we propose a simple way to identify the movements performed in one step (one game move).

Assume that an aircraft is located at point A having velocity v_0, see Figure 4.

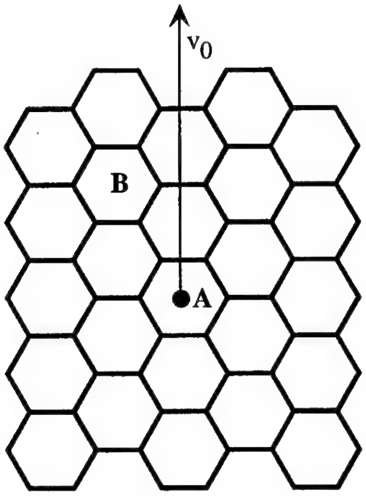


Figure 4. Initial piece position and velocity

1. We would like to find out whether we can move it to point B in one game move. In addition, we would like to maintain constant speed, say $v = |v_0|$, but we may vary directions. If that were possible in some smooth, safe way (no crazy maneuvers allowed), we would also like to find the velocity vector v_1 at the destination.
2. Assume further that r_0 is the radius of the tightest safe turn the aircraft could perform at speed v and that all the game moves have a fixed duration t.
3. In order to use the simplistic approach, we will do the following:

3.1. Draw a circle tangential to the vector v_0 at the point A so that it would intersect point B. Assume that its radius is r_1;

3.2. Draw a vector v_1 tangential to this circle at the point B and such that $|v_1| = |v_0|$. Assume that the arc length between A and B is equal to d.

4. Then B is reachable by this aircraft with the end velocity v_1 if $r_0 \leq r_1$ and $d = v*t$.

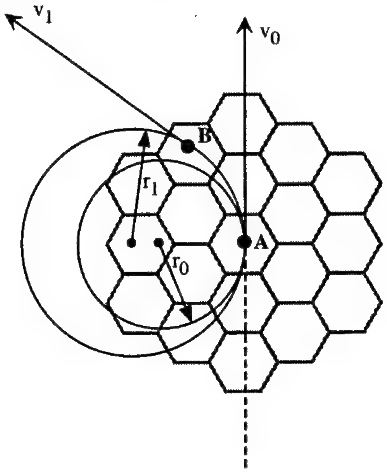


Figure 5. Computing motion reachability

2.4.2 Weapon Reachability

In order to model the ability to destroy targets with long range weapons, we have to define an additional class of reachability relation called weapon reachability relations. Figure 6 illustrates this concept for a long-range missile with effective radius of 20 miles. Each hex's inner circle has diameter of 2 miles. The Red Fighter direction is as shown in the figure. The reachable hexes are shown in light blue (or in light gray if rendered in black and white). The obstacles are in black. The hexes shielded by the obstacles are dark gray. Notice, that if **X** be the Board and **D** be the set of admissible directions, then a Motion Reachability would be a relation on **(X×D)×(X×D)**, whereas a Weapon Reachability would be a relation on **(X×D)×X.**

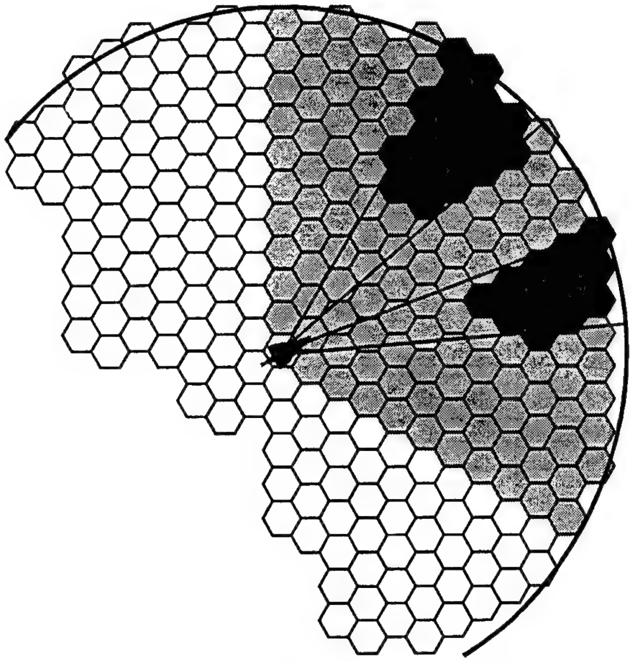


Figure 6. Long Range Missile reachability relation

3 Strategy Generation with Trajectories and Zones

3.1 Formalizing Abstract Board games within LG

We shall now describe a formal representation of a multi-agent system as an abstract board game. Informally, such game would define a "discrete universe" by observing "the laws of discrete physics". The problems in such universe are very close to the board games like chess, checkers, etc. An abstract board or an area of the discrete universe, is represented by a suitable finite set **X**. Abstract pieces (also called the elements), represent the local agents staying in place or moving with variable speeds. We introduce such actions as the movement of agents, the destruction of agents, utilization of a long range weapon, collision, collision avoidance, etc. For the alternating concurrent (AC) and totally concurrent (TC) systems, we introduce concurrent movements and actions.

DEFINITION 1. An ABG is the following ten-tuple:

$$\langle \mathbf{X}, \Omega, \mathbf{P}, \mathbf{R}, \mathbf{SPACE}, val, \mathbf{S}_{in}, \mathbf{S}_f, \mathbf{TR}, \mathbf{W} \rangle,$$

where

X = the game Board, represented as a finite set $\{x_i\}$ of *locations*. Depending on the problem domain, there may be different kinds of (abstract) locations. For instance, in two-dimensional space, in three-dimensional space, or in a phase space, e.g., as pairs of 3D space locations and directions of motion. Notice that sometimes there may be several equivalent representations of the game Board and pieces (see below). For instance, instead of considering the phase space as above, the Board may be the 3D space, whereas the direction component may be assigned to a piece as a part of its state;

$\Omega = \{\omega_1, ..., \omega_n\}$ is the set of *players*, (also called *sides*). Often Ω consists of two players, ω_1 and ω_2, called the *opposing sides*. In our initial examples, we'll mostly assume two opposing sides;

$\mathbf{P} = \{P_1, ..., P_n\}$ is the assignment of *pieces*. Each P_i is the set of *pieces* assigned to the player ω_i. This assignment must be disjoint. Each piece may possess a state, e.g., the velocity or the direction of motion for aircraft;

R is a set of relations of *reachability* (e.g., *motion* or *weapon* reachability) in X. Such relations may have signatures consisting of various combinations of the Board and components of the piece state space. For instance, for aircraft we may employ the motion reachability relation $\mathrm{MR} \subseteq (\mathbf{X} \times \mathbf{D}) \times (\mathbf{X} \times \mathbf{D})$ as well as a weapon reachability relation $\mathrm{WR} \subseteq (\mathbf{X} \times \mathbf{D}) \times \mathbf{X}$ where **D** is the space of directions. Notice that in general the reachability relations depend on the game state (see below). However, in this report we assume that there is no such dependency. We will introduce such dependency when necessary via formalization of the notion of "dynamic obstacle" (see [11]).

val is a function on P with positive integer values describing the initial *values* of elements;

SPACE is the game state space. A state $s \in \mathbf{SPACE}$ consists of a partial function of *placement* of the form ON_s: $\mathbf{P} \to \mathbf{X}$ and of some additional parameters.

The value $\mathrm{ON}_s(p) = x$ means that element p occupies location x at state S. When there is no confusion we will write ON instead of ON_s. To describe the function ON for a state s, we may write equations, called the placement equations, of the form $\mathrm{ON}(p) = x$ for all elements p where ON is defined. Notice that the placement function restricts possible legitimate placements of the pieces on the Board. In common board game this function is defined via game rules.

The additional parameters of a state may be various finite automata. E.g., the state of an *alternating* Complex System may include a two-valued automaton describing the player whose turn is to move at the state.

$\mathbf{S}_{in}$, $\mathbf{S}_f \subseteq \mathbf{SPACE}$ are the sets of permitted *initial* and *final* states, respectively.

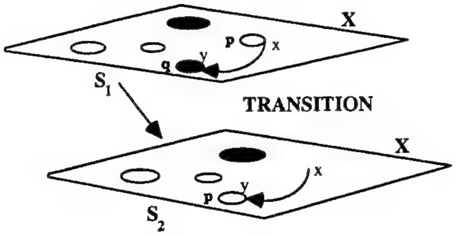


Figure 7. A state transition: piece p moves from x to y while eliminating piece q

TR$\subseteq$**SPACE**$\times$**SPACE** is the set of all permissible state transitions, see Figure 7. There are various means to describe the state transitions. Usually it is done via a collection of the game rules. E.g., a rule can be described by (1) the guard describing the applicability of the rule to the source state; (2) the *remove list* consisting of the placement equations of the pieces to be removed from the board; and (3) the *add list* consisting of the placement equations of the pieces to be added to the board. Finally, the guard may be described via an *applicability list* consisting of items pertaining to the individual pieces, e.g., of the form ON(p) = $x \wedge R_p(x,y)$. **TR** permits us to define the notion of a *state run* (also called a *play*). A play is a sequence of states $s_0s_1...s_k$ such that $s_i \in$ **SPACE**$-$**S**$_f$ and $(s_i,s_{i+1}) \in$ **TR** for i = 0, ..., k−1, $s_0 \in$ **S**$_{in}$, and $s_k \in$ **S**$_f$. We designate the set of all such plays as **PLAY**;

W = {W_1, ..., W_n} is the assignment of winning sets for every player. Each W_i is the set of winning plays assigned to the player ω_i in the sense that the player wins a play if the play is in W_i, and looses otherwise. The above assignment is not necessary exclusive, i.e., it is possible for more than one player to win.

■

The winning sets can be defined in various ways. Consider an example. Let Ω = {ω_1, ω_2} and let **S**$_f$ = $S_1 \cup S_2 \cup S_3$. We define W_i for i = 1, 2 as the set of all plays with a final state in either S_i or in S_3. We may call the states from S_i the wins for ω_i and the states from S_3 the draw states. Informally, the *goal* of each side is to reach either its winning state or a draw state. Let us designate ω_1 as the Friend. Then the problem of controlling this System may be represented as a problem of searching for a sequence of transitions leading from an initial state in S$_{in}$ to a final state in $S_1 \cup S_3$.

3.2 The LG Tools

Construction of the LG strategies involves several tools of increasing sophistication:

- Distance measuring tools
- Trajectory generating tools
- Zone generating tools
- LG strategy tree generating tools

In this Report, we'll discuss only the first three categories of tools. They are the essential ingredients of the methodology. We have modified and enhanced the original definitions and algorithms from [11] to make them suitable to the Air Operations domain. The complete approach is described in [11]. However, there are various possible extensions and subsets of the LG strategy tree generating tools described in [11] suitable to the Air Operations domain. Although we implemented some of them in our software, we'll formally describe them in a future report.

DEFINITION 2. (Extended Board) In order to formulate the needed adaptations of the LG concepts to the Air Operations domain, we need an addendum to the ten-tuple defined earlier. We designate as **D** the set of all possible legal directions where a piece may move. At this point, it is irrelevant how this set is defined. An example of such definition is illustrated in Figure 8. We'll sometimes refer to **X**×**D** as an extended Board and we'll call the elements of **X**×**D** the extended locations.

■

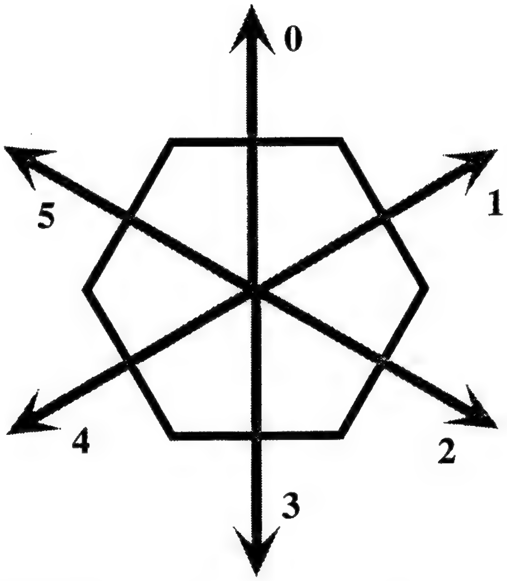


Figure 8. Defining directions with respect to a hex cell

3.3 Computing Distances and Avoiding Obstacles

The LG mechanism to avoid obstacles is based on two functions, MAP:(**X**×**D**)×(**X**×**D**)→**N** and SMAP: (**X**×**D**)×**X**→**N**, where SMAP stands for "Strike MAP" and **N** is the set of natural numbers. The definitions and the LG algorithm for the MAP and SMAP functions are given below. These functions extend the MAP given in [11] by taking into account the velocities represented by the direction of movement, as well as the weapon reachability relations. Let MR$\subseteq$(**X**$\times$**D**)$\times$(**X**$\times$**D**) be a motion reachability relation, and let WR$\subseteq$(**X**$\times$**D**)$\times$**X** be a weapon reachability relation. The function MAP will be parameterized by MR, whereas SMAP will be parameterized by both MR and WR. Note that in this section the letters *x, y, z* are not intended to represent x-y-z coordinates, but locations on the Board in some numbering system.

DEFINITION 3. (MAP and SMAP)

- For each piece *p*, 3D locations *x*, *y*, and directions *a*, *b*, $\text{MAP}_p((x,a), (y,b))$ is the length of the shortest trajectory, see DEFINITION 4, for the piece *p* to travel from the location (x,a) to the location (y,b). When there is no confusion, we'll abbreviate $\text{MAP}_p((x,a), (y,b))$ as MAP$((x,a), (y,b))$.
- For each piece *p*, weapon kind *wk*, 3D locations *x, y, z*, and directions *a, b, c*, $\text{SMAP}_{p,wk}((x,a), y)$ is the length of the shortest trajectory for the piece *p* to travel from the location (x,a) to some location (z,c) so that the location *y* would be within the range of the weapon of kind *wk*. When there is no confusion, we'll abbreviate $\text{SMAP}_{p,wk}((x,a), y)$ as SMAP$((x,a), y)$.

■

THEOREM 1. Metric properties of MAP and SMAP.

- Definition of MAP. For each piece location *x, y, z*, and direction *a, b, c*:
  - MAP$((x,a), (y,b)) \geq 0$;
  - MAP$((x,a), (x,a)) = 0$;
  - if MR$((x,a), (y,b))$ then MAP$((x,a), (y,b)) = 1$;
  - MAP$((x,a), (y,b))$ + MAP$((y,b), (z,c)) \geq$ MAP$((x,a), (z,c))$.
- Definition of SMAP. For each location *x, y, z*, and direction *a, b*:
  - SMAP$((x,a), y) \geq 0$;
  - if WR$((x,a), y)$ then SMAP$((x,a), y) = 0$;
  - if SMAP$((x,a), y) \neq 0$ and if there exists some (w,d) with MR$((x,a), (w,d))$ and WR$((w,d), y)$, then SMAP$((x,a), y) = 1$;
  - MAP$((x,a), (y,b))$ + SMAP$((y,b), z) \geq$ SMAP$((x,a), z)$.

■

Informally, MAP$((x,a), (y,b))$ is the minimal number of steps the aircraft could move from a location *x* with an initial direction *a* to a location *y* with a final direction *b*. SMAP$((x,a), y)$ is the minimal number of steps the aircraft could move from a location *x* with an initial direction *a* to some arbitrary location and direction from which it can fire a weapon that would strike a target at *y*.

In order to describe the algorithm computing MAP and SMAP, we need two auxiliary functions, Move_Space and Strike_Space:

- Move_Space$(x,a) = \{(y,b) \in \mathbf{X}\times\mathbf{D} \mid \text{MR}((x,a), (y,b))\}$
- Strike_Space$(x,a) = \{y \in \mathbf{X} \mid \text{WR}((x,a), y)\}$

It will also be convenient to promote the above functions to the operations on the subsets of **X**$\times$**D**, so that if $V \subseteq \mathbf{X}\times\mathbf{D}$ then:

- Move_Space$(V) = \cup\{\text{Move_Space}(y,b) \mid (y,b) \in V\}$
- Strike_Space$(V) = \cup\{\text{Strike_Space}(y,b) \mid (y,b) \in V\}$

Our algorithm, see Table 1, is written in an extension of the Dijkstra language, see [1]. The algorithm is based on the MAP generating algorithm from [11] and is modified to include SMAP necessary for the Air and other military operations.

3.4 Trajectories.

3.4.1 Defining Trajectories

Informally, a trajectory is a path on the Board or the Extended Board, with an entity moving along it. When the entity moves forward, part of a trajectory that is left behind may disappear, and re-appear again, when an entity would backtrack during the search to explore another path. In LG, the trajectories are represented as strings of symbols.

DEFINITION 4. A *trajectory* from $x \in \mathbf{X}\times\mathbf{D}$ to $y \in \mathbf{X}\times\mathbf{D}$, for a piece $p \in \mathbf{P}$, of *length l*, in the Extended Board is a string of the form $t = \boldsymbol{a}(x_0)\boldsymbol{a}(x_1) \ldots \boldsymbol{a}(x_l)$ over the alphabet $\{\boldsymbol{a}\}\times\mathbf{X}\times\mathbf{D}$ and such that:

- $x = x_0$ and $y = x_l$;
- $\text{MR}_p(x_i, x_{i+1})$ holds for $i = 0, 1, \ldots, l-1$, where MR_p is a motion reachability.

With respect to this trajectory, we call *x* the trajectory extended source and *y* the trajectory extended destination. The projections of *x* and *y* onto the Board we'll call, respectively, the source and the destination. We call the extended locations $x_0, x_1, \ldots, x_l$ the trajectory nodes. The set of all trajectories for a piece *p* of the length less than or equal **H** is called the Language of Trajectories within horizon **H**. We designate it as $\mathbf{L}_t^{\mathbf{H}}(p)$. The set of all trajectories for a piece *p* of the length equal **H** is called the Strict Language of Trajectories within horizon **H**. We designate it as $\mathbf{SL}_t^{\mathbf{H}}(p)$.

■

DEFINITION 5. A *strike trajectory* from $x \in \mathbf{X}\times\mathbf{D}$ to $y \in \mathbf{X}$, for a piece $p \in \mathbf{P}$, of *length l*, in extended Board is a trajectory $t = \boldsymbol{a}(x_0)\boldsymbol{a}(x_1) \ldots \boldsymbol{a}(x_l)$ such that:

- $x = x_0$ and $y = x_l$;
- $\text{WR}_p(x_l, y)$ holds, where WR_p is a weapon reachability.

With respect to this strike trajectory, we call *y* the strike trajectory target location. The set of all strike trajectories for a piece *p* of the length less than or equal **H** is called the Language of Strike Trajectories within horizon **H**. We designate it as $\mathbf{L}_{st}^{\mathbf{H}}(p)$. The set of all strike trajectories for a piece *p* of the length equal **H** is called the Strict Language of Strike Trajectories within horizon **H**. We designate it as $\mathbf{SL}_{st}^{\mathbf{H}}(p)$.

■

Formal tools, i.e., grammars of trajectories [11], generate various kinds of trajectories encountered in a number of problem domains, e.g., shortest trajectories, admissible trajectories (that are concatenations of two shortest trajectories), etc., see Figure 9. In this report we'll concentrate on the shortest trajectories and strike trajectories.

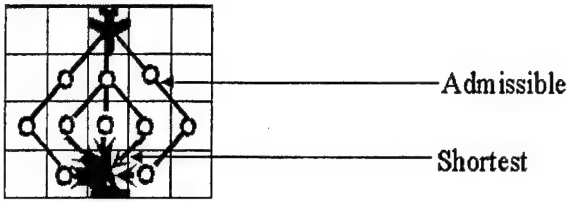


Figure 9. Shortest and admissible trajectories

3.4.2 Generating Bundles of Shortest Trajectories

We define a bundle of shortest trajectories, as the set of all the shortest trajectories between two locations on the extended Game Board. Notice that since we are dealing with obstacles, a shortest trajectory is not necessarily a straight line. We represent a bundle of trajectories as a directed graph, called the bridge of the bundle, where nodes are the extended locations, each representing a trajectory node, and the edges are the moves along the trajectories (see Figure 10). We represent a bridge as a sequence of *layers*. A layer of distance *n* is the collection of all the nodes from the bridge, each being the n^{th} trajectory node for one the trajectories in the bundle. Such sequence of layers is generated by the controlled grammar ShortestBundle in Table 2, also see [11,15].

DEFINITION 6. The set of all trajectory bundles for a piece *p* of the length less than or equal **H** is called the Language of Trajectory Bundles within horizon **H**. We designate it as $\mathbf{L}_{tb}^{\mathbf{H}}(p)$. The set of all trajectory bundles for a piece *p* of the length equal **H** is called the Strict Language of Trajectory Bundles within horizon **H**. We designate it as $\mathbf{SL}_{tb}^{\mathbf{H}}(p)$.

■

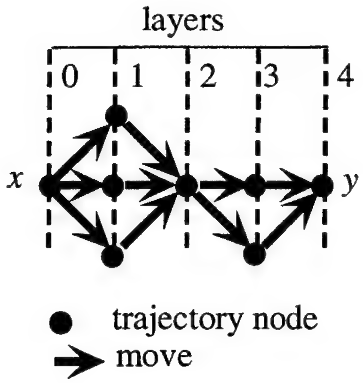


Figure 10. A bundle of shortest trajectories

3.4.3 Generating Bundles of Shortest Strike Trajectories

We define a bundle of shortest strike trajectories, as the set of all the shortest strike trajectories between an extended location and a location on the Game Board. In order to depict such structure, we need to project it from the extended Board onto the Board, see Figure 11. The generation grammar is given in Table 3. It is similar to the grammar for the shortest trajectory bundles, but has several subtle differences stemming from usage of SMAP and the fact that the target is not necessarily a part of a trajectory on the Extended Board, see Figure 11.

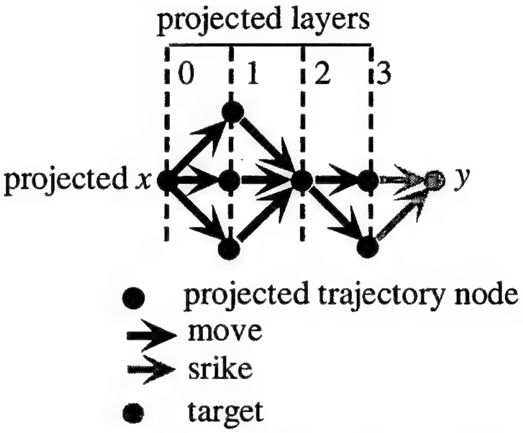


Figure 11. A bundle of shortest strike trajectories

DEFINITION 7. The set of all strike trajectory bundles for a piece *p* of the length less than or equal **H** is called the Language of Strike Trajectory Bundles within horizon **H**. We designate it as $\mathbf{L}_{stb}^{\mathbf{H}}(p)$. The set of all strike trajectory bundles for a piece *p* of the length equal **H** is called the Strict Language of Strike Trajectory Bundles within horizon **H**. We designate it as $\mathbf{SL}_{stb}^{\mathbf{H}}(p)$.

■

3.5 Zones

The notion of a Zone is crucial for generating LG strategies. There are several kinds of LG Zones, e.g., *attack*, *retreat*, *unblock*, etc., see [11]. In this report we'll concentrate on the attack Zones but will add a new notion of a bundle of attack Zones. In addition, in contrast with [11], our Zones will employ the strike trajectories and their bundles in addition to the trajectories.

Roughly speaking, an attack "Zone" (Figure 12) has a main (friendly or adversary) piece (e.g., $\mathbf{p}_0$ on Figure 12) and a main strike trajectory (e.g., 1,2,3,4,(5)), that is a path that the main agent needs to attain a local goal. A Zone includes a number of opposing pieces (e.g., $\mathbf{q}_0$, $\mathbf{q}_3$) and their strike trajectories (e.g., 6,7,8,(9)) capable of preventing the main agent from achieving the goal. It also includes auxiliary friendly pieces counteracting the above actions of the enemy, counter-counteractions of the opponents, etc. The continuous lines indicate the directions of physical moves, whereas the dashed lines indicate the action, that is the weapons release in the example on Figure 12.

The Zone shown in Figure 12 has 3 aircraft for the red side and 3 aircraft and a tank for the blue side. Therefore, the pieces are the 6 aircraft and the tank. With respect to this Zone, the aircraft are intended to move along the indicated trajectories. The small circles along the trajectories indicate the possible moves (trajectory nodes). For example, the red aircraft, $\mathbf{p}_0$, will have the Blue Tank $\mathbf{q}_0$ in shooting range in three moves, once the aircraft moves along its trajectory 1, 2, 3, 4. However, the blue aircraft $\mathbf{q}_1$, and $\mathbf{q}_2$ can get the position 3 into their shooting range once they reach positions 8 and 11, respectively. The blue aircraft $\mathbf{q}_2$ and $\mathbf{q}_3$ can be prevented from reaching their respective destinations by the red aircraft $\mathbf{p}_1$ and $\mathbf{p}_2$. Finally, the blue aircraft $\mathbf{q}_3$ can effectively prevent $\mathbf{p}_0$ from achieving its goal. Thus, the red side would not win this local combat. Therefore, with respect to this Zone the piece $\mathbf{p}_0$ is at disadvantage.

A Zone is strict if the length of any negation trajectory *t* is equal to the number of moves that the acting piece on the negated by *t* trajectory has to make for reaching the target location of *t*. To think of a bundle of Zones, it is enough to replace each strike trajectory *t* in a singleton Zone with a bundle of strike trajectories with the same source and target as those of *t*.

In order to use the Zone in the Air operation domain, we need a controlled grammar generating bundles of strict attack Zones. It is described in Table 4.

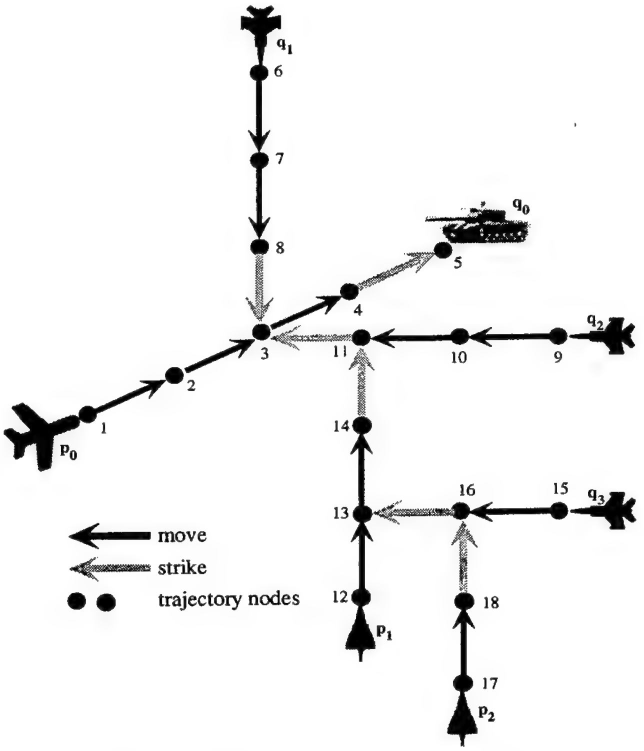


Figure 12. A simple attack Zone

4 Comparison with other Approaches

4.1 Summary of the Comparison of LG and other Approaches

- In our approach, application of the control theory (DES + HS) is subordinate to the application of the game theory (LG) in the following sense. The LG part is responsible for selection of the overall strategy and the local strategies spanning over several moves, whereas the control part is responsible for making each single move in compliance with the LG overall and local strategies and for automatic reaction to certain events in between the LG moves;
  - In many other approaches, the gaming parts (if present) have subordinate roles. This makes it difficult to select an overall winning strategy as well as local winning strategies;
- We utilize a class of extended concurrent games called Abstract Board Games (ABG). It permits us to explicitly envision the Battle Theater as a game Board. It also permits us explicit manipulation with battle units as the game pieces. The actions of the adversary are also explicitly represented;
  - In many other approaches, normalized games are used. Then the battlefield and the battle units are

encoded as coefficients in some equations, whereas the adversary is seen as "disturbance". This makes it difficult, if not impossible, to faithfully model the battlefield theater and units. Seeing the adversary as disturbance would allow one to model only very gross enemy behaviors;

- We utilize a new type of game theory for solving ABG called Linguistic Geometry (LG). Although, at present time we don't have a mathematical proof that the method will always produce a winning or optimal strategy, such proofs exist for several classes of games, [11]. For other classes of games, past experiments demonstrated that LG yields near-optimal strategies sufficient to produce solutions, whereas the other methods failed to do so (the power plant project at USSR Ministry of Energy, etc., [11]).
  - In many other approaches, various techniques for solving normalized games are used. Some of these techniques have exponential complexity, whereas others would solve games with only few units an simplest enemy representation;
- Horizontal scalability of LG allows us to control multiple 3D local combats simultaneously;
  - In many other approaches, only 2D representations are modeled;
- Vertical scalability of LG allows us to support many levels of command and control: tactical, operational, and strategic. This is because we can represent the abstract game board at various levels of abstraction. Mappings may be defined between various levels and simultaneous coordinated games may be run if desired, see Figure 15.
  - In many other approaches, such mappings are very difficult to achieve since the entities are represented as coefficients or parameters.

4.2 JFACC Games

4.2.1 The role of the Gaming Part: Subordinate vs. Leading

It is widely accepted that control theory by its nature (at its current status) cannot handle intelligent adversary. In control theory, adversarial actions are usually considered as random perturbations, which is inadequate for modeling hostile counteractions. All the JFACC teams that use control theory are forced to apply various types of game theory to handle intelligent adversaries. This application is subordinate to their respective version of control theory. The Rockwell Team utilizes control theory within the group of mobile entities (aircraft, SAMs, etc.). In our approach the global strategy of Blue and Red forces is planned and controlled by the game theory and verified locally by the control theory.

4.2.2 Extended vs. Normalized Games

The games used by the JFACC teams (except for Rockwell Team) are *normalized* games or one-step games. They were introduced and investigated by Von Neumann and Morgenstern half a century ago and later developed by multiple followers. This approach allows analyzing full game strategies, representing entire games. It does not allow breaking a game into separate moves and comparing them. Only full strategies, the entire courses of behavior of players can be compared. This significant limitation makes this approach inadequate for real world C2 problems. Von Neumann-Morgenstern games and respective strategies have been represented in the discrete or continuous (differential) form. For both types of games, discrete and differential, advanced theoretical results have been received. However, these advanced theories lack scalability. The classic approaches based on the conventional theory of differential games are insufficient, especially in case of dynamic, multi-agent models. It is well known that there exist a small number of differential games for which exact analytical solutions are available. There are a few more differential games for which numerical solutions can be computed in a reasonable amount of time, albeit under rather restrictive conditions. However, each of these games must be one-to-one, which is very far from the real world combat scenarios. They are also of the "zero-sum type" which does not allow the enemy to have goals other than diametrically opposing to those of the Friend. Other difficulties arise from the requirements of the 3D modeling, limitation of the lifetime of the agents, or simultaneous participation of the heterogeneous agents such as on-surface and aerospace vehicles.

Another class of games is called *extended* games. Out team utilizes this class of games. Extended games are usually represented as trees, which include every alternative move of every strategy of every player. Application of this class of games to real world problems requires discretization of the domain, which can be done with various levels of granularity. In addition, in the real world problems, moves of all the pieces (aircraft) and players (Red and Blue) are concurrent, and this can be represented within extended, but not within normalized, games. Thus, the extended games would allow us to adequately represent numerous problem domains including military C2. The main difficulty for any game approach is the "curse of dimension." Even for a small-scale combat, an extended game may be represented by a game tree of astronomic size, which would make this game intractable. Even the most presently promising search algorithms on the game trees, those that utilize alpha-beta pruning, result in search reduction, which is still insufficient. Even in the best case the number of moves to be searched employing alpha-beta algorithms grows exponentially with the power of this exponent divided by two with respect to the original game tree. The alpha-beta pruning method is applicable for sequential alternating games only (Blue-Red-Blue-…), whereas most of the real world games, including Air and other military operations are concurrent. For the games with concurrent actions the number of moves to be searched "explodes" even more dramatically than for the sequential games. This is because of all the possible combinations of moves for different pieces that can be included in one concurrent move. With conventional approaches, the question of scalability of extended concurrent games cannot be even raised.

The new type of game theory, LG, allows us to overcome these obstacles.

5 LG-ABG Experiments

The theoretical computational complexity associated with the LG was not yet evaluated. However there is significant empirical evidence that the complexity is low end polynomial, closer to linear. The LG approach was implemented within the DARPA JFACC project and a number of experiments were conducted, see [4]. Some of the empirical evidence collected so far is represented on Figure 13 and Figure 14. The complexity of the Zone structure is proportional to the number of trajectory bundles and to the structure size in KB. Ten zones of various complexities were generated for these experiments. One of these Zones is illustrated on Figure 17.

To generate a full LG strategy with each of these 10 scenarios for which the above 10 initial Zones were generated would require about 40 to 60 moves, depending on the length of the main trajectory of the initial Zone. A new Zone would be generated for each move, so that total time would be about 30 min.

In contrast, to generate a complete strategy for these scenarios with another game methodology, alpha-beta pruning would not be feasible. Indeed, for each of our scenarios at least six aircraft were involved. Each of these can choose any out of 18 possible moves at each step. Thus, at least 40 moves would give the size of the unreduced search tree as $(18^6)^{40} = 18^{240}$. The theoretical maximal reduction of the alpha-beta pruning method is the square root of the unreduced tree, which gives us 18^{120}.

6 Conclusion

The LG approach is a new revolutionary methodology. It permits us to find solutions for the game problems for which alternative methodologies such as alpha-beta pruning are unable to provide any solution. In addition, the methodology is adaptable to various domains including Air and other military operations.

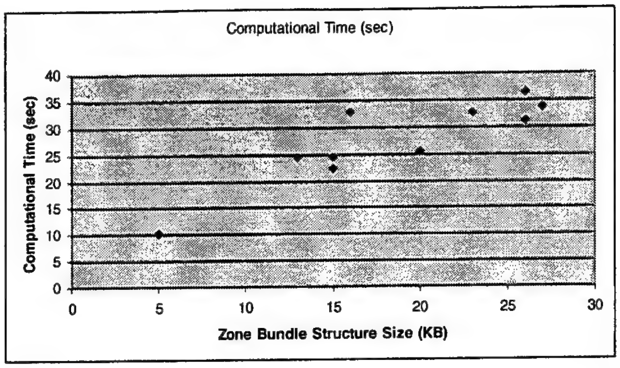


Figure 13. Computational time vs. size for Zone bundles

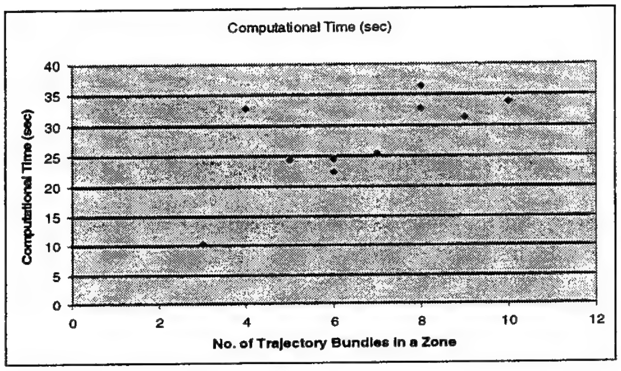


Figure 14. Computational time vs. number of trajectory bundles for Zone bundles

Bibliography

1 D. Gries. *The Science of Programming*, Springer-Verlag, 1981.

2 R. E. Fikes and N. J. Nilsson. STRIPS: A New Approach to the Application of Theorem Proving in Problem Solving, *Artificial Intelligence*, 2:189--208, 1971.

3 D. E. Knuth and R. W. Moore. An Analysis of Alpha-Beta Pruning, Artificial Intelligence, 6(4):293--326, 1975.

4. J. Lee, Y.-L. Chen, V. Yakhnis, B. Stilman and F. Lin. DARPA JFACC Experiment Plan. Technical Report, Rockwell Science Center, 2000.

5 N. J. Nilsson. *Principles of Artificial Intelligence.* Tioga Publ., Palo Alto, CA, 1980.

6 E. D. Sacerdoti. The Nonlinear Nature of Plans. In *Proc. Int. Joint Conference on Artificial Intelligence*, 206--214, 1975.

7 B. Stilman. A Linguistic Geometry for Control Systems Design. *Int. J. of Computers and Their Applications*, 1(2):89--110, 1994.

8 B. Stilman. Managing Search Complexity in Linguistic Geometry. *IEEE Transactions on Systems, Man, and Cybernetics*, 27(6):978--998, 1997.

9 B. Stilman. Network Languages for Concurrent Multi-agent Systems. *Intl. J. of Computers & Mathematics with Applications*. 34(1):103--136, 1997.

10 B. Stilman and C. Fletcher. Systems Modeling in Linguistic Geometry: Natural and Artificial Conflicts. *Intl. J. of Systems Analysis, Modeling, Simulation*, 33:57--97, 1998.

11 B. Stilman. *Linguistic Geometry: From Search to Construction*. Kluwer Acad. Publishers, 2000.

12 B. Stilman and V. Yakhnis. Solving Adversarial Control Problems with Abstract Board Games and Linguistic Geometry (LG) Strategies, DARPA-JFACC 1st AEC SYMPOSIUM, San Diego, CA, November 15-16, 1999

13 A. Yakhnis and V. Yakhnis. Gurevich-Harrington's Games Defined by Finite Automata. *Annals of Pure and Applied Logic*, 62:265--294, 1993.

14 V. Yakhnis. Grammar of Zones with Admissible Trajectories of Arbitrary Degrees, 6pp, IEEE SMC'98, 1998.

15 V. Yakhnis and B. Stilman. A Multi-Agent Graph-Game Approach to Theoretical Foundations of Linguistic Geometry. In *Proc. of the Second World Conference on the Fundamentals of Artificial Intelligence* (WOCFAI 95), Paris, France, July, 1995.

16 S. Zeitman. Unforgettable forgetful determinacy. *Logic and Computation*, 4:273--283, 1994.

Appendix A. Tables

Table 1. The algorithm for generation of MAP and SMAP functions

```
algorithm  MAP and SMAP Generation
purpose  Let X be the Board and D be the set of admissible directions. Given a piece $p$ with motion reachability relation R, an initial location $x \in \mathbf{X}$ and initial direction $a \in \mathbf{D}$, for each legal location $y \in \mathbf{X}$ and direction $b \in \mathbf{D}$ generate MAP((x,a), (y,b)) and SMAP((x,a), y)
precondition  (x,a) is a legal pair for p
declarations
  $U, V : Power(\mathbf{X} \times \mathbf{D})$ /* Power() is the power set operator. */
  $W, Y : Power(\mathbf{X})$
  $n : \mathbf{N}$
end
$U, V := \{(x, a)\}$
$W$ = Strike_Space($x,a$)
$n := 0$
MAP((x,a), (x,a)) := 0
for all $y \in W$ do SMAP((x,a), y) := 0 od
do
  Move_Space(V)$-U \neq \emptyset$ →
      $V :=$ Move_Space(V)$-U$
      $U := U \cup V$
      $Y :=$ Strike_Space(V)$-W$
      $W := W \cup Y$
      $n := n + 1$
      for all $(y,b) \in V$ do MAP((x,a), (y,b)) := n od
      for all $y \in Y$ do SMAP((x,a), y) := n od
  invariant
    $V \subseteq U \wedge \{(x, a)\} \times U \subseteq$ dom(MAP) $\wedge$ Move_Space($U-V$)$\subseteq U$
    $\wedge$
    $Y \subseteq W \wedge \{(x, a)\} \times W \subseteq$ dom(SMAP) $\wedge$ Strike_Space($U-V$)$\subseteq W$
  variant
    #(X×D)
od
for all $(y,b) \in \mathbf{X} \times \mathbf{D} - U$ do MAP((x,a), (y,b)) := ∞ od
for all $y \in \mathbf{X} - W$ do SMAP((x,a), y) := ∞ od
```

Table 2. The grammar of shortest trajectory bundles

Grammar ShortestBundle($p : \mathbf{P}, x, y : \mathbf{X}{\times}\mathbf{D}$)

declarations	**definitions**
$u, v, w : \mathbf{X}{\times}\mathbf{D}$	MR, MAP are given with respect to piece p
U, V : Power($\mathbf{X}$)	$K \equiv \text{MAP}(x,y)$
$K, l : \mathbf{N}$	$\text{NextBetween}(u,w) \equiv \text{MR}(u,w) \wedge \text{MAP}(x,u)+1+\text{MAP}(w,y) = \text{MAP}(x,y)$
	$\text{Occur}(\text{A}(U,V,l)) \equiv$ the symbol $\text{A}(U,V,l)$ occurs in the current string
	$\#\text{Occur}(\text{A}(U,V,l)) \equiv$ the number of occurrences of symbol $\text{A}(U,V,l)$ in the current string
	corollary
	$\text{NextBetween}(u,w) \Rightarrow$ trajectory uw is a sub-trajectory of a shortest trajectory between x and y

do

invariant

$\#\text{Occur}(\text{A}(U,V,l)) \leq 1 \wedge \#\text{Occur}(\text{a}(U)) \leq 1 \wedge (\text{Occur}(\text{A}(U,V,l)) \Rightarrow (\forall u \in U \bullet l = \text{MAP}(u,y)) \wedge (\forall v \in V \exists u \in U \bullet \text{NextBetween}(u,v)))$

corollary

$\text{Occur}(\text{A}(U,V,1)) \Rightarrow V = \varnothing$

guard **true**	$\text{I} \rightarrow \text{A}(\{x\},\varnothing,\text{K})$
guard $l{>}1 \wedge w \in \mathbf{X}-V \wedge (\exists u \in U \bullet \text{NextBetween}(u,w))$ **variant** $\#(\mathbf{X}-V)$	$\text{A}(U,V,l) \rightarrow \text{A}(U,V \cup \{w\},l)$
guard $l{>}1 \wedge (\forall w \in \mathbf{X}-V \forall u \in U \bullet \neg\text{NextBetween}(u,w))$ **variant** l	$\text{A}(U,V,l) \rightarrow \text{a}(U)\text{A}(V,\varnothing,l-1)$
guard $V = \varnothing \wedge l{=}1$ **od**	$\text{A}(U,V,l) \rightarrow \text{a}(U)^*\text{a}(\{y\})$

Table 3. The grammar of shortest strike trajectory bundles

Grammar ShortestStrikeBundle($p : \mathbf{P}, x : \mathbf{X}{\times}\mathbf{D}, y : \mathbf{X}$)

declarations	**definitions**
$u, v, w : \mathbf{X}{\times}\mathbf{D}$	WR, MAP, SMAP are given with respect to piece p
U, V : Power($\mathbf{X}{\times}\mathbf{D}$)	$K \equiv \text{SMAP}(x,y)$
$K, l : \mathbf{N}$	$\text{NextBeforeStrike}(u,w) \equiv \text{WR}(u,w) \wedge \text{MAP}(x,u)+1+\text{SMAP}(w,y) = \text{SMAP}(x,y)$
	$\text{Occur}(\text{A}(U,V,l)) \equiv$ the symbol $\text{A}(U,V,l)$ occurs in the current string
	corollary
	$\text{NextBeforeStrike}(u,w) \Rightarrow$ trajectory uw is a sub-trajectory of a shortest strike trajectory between x and y

do

invariant

$\text{Occur}(\text{A}(U,V,l)) \Rightarrow (\forall u \in U \bullet l = \text{SMAP}(u,y)) \wedge (\forall v \in V \exists u \in U \bullet \text{NextBeforeStrike}(u,v))$

corollary

$\text{Occur}(\text{A}(U,V,1)) \Rightarrow V = \varnothing$

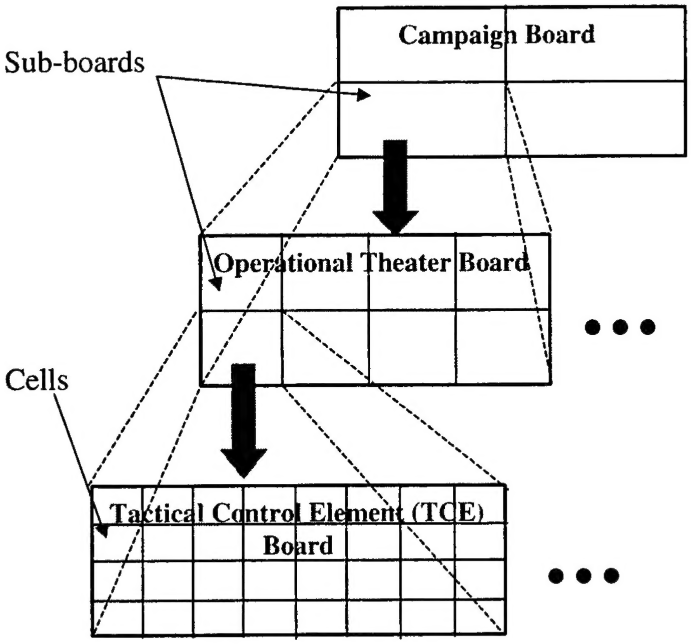
guard **true**	$\text{I} \rightarrow \text{A}(\{x\},\varnothing,\text{K})$
guard $l{>}1 \wedge w \in \mathbf{X}-V \wedge (\exists u \in U \bullet \text{NextBeforeStrike}(u,w))$ **variant** $\#(\mathbf{X}-V)$	$\text{A}(U,V,l) \rightarrow \text{A}(U,V \cup \{w\},l)$
guard	$\text{A}(U,V,l) \rightarrow \text{a}(U)\text{A}(V,\varnothing,l-1)$

$l>1 \wedge (\forall w \in \mathbf{X}-V \forall u \in U \bullet \neg \text{NextBeforeStrike}(u,w))$ **variant** l	
$l=1$ **od**	$\text{A}(U,\varnothing,l) \rightarrow \text{a}(U)$

Table 4. The grammar of shortest Zone bundles

Grammar ShortestZoneBundle(p : $\mathbf{P}$, x : $\mathbf{X}\times\mathbf{D}$, y : $\mathbf{X}$; n : $\mathbf{N}$)	
declarations $p, q : \mathbf{P}$ $r, t : \mathbf{L}_{\text{stb}}^{\text{H}}(p)$ $l : \mathbf{N}$	**definitions** Vulnerable(q, t) ≡ the weapon utilized at t may be applied against q Actor(t) ≡ the piece moving along t New(r) ≡ r is not present in the sequence generated so far StrictAttack(r, t) ≡ TargetLocation(r)∈Nodes(t) ∧ Length(r) = TrajectoryDistance(TargetLocation(r),t) ∧ Vulnerable(Actor(t),r) ∧ Opposing(Actor(r),Actor(t)) Λ ≡ empty string
do **invariant** $\text{Occur}(\text{B}(t, l)) \Rightarrow l \leq n$	
guard **true**	$\mathit{I} \rightarrow$ a(ShortestStrikeBundle(p, x, y), 0)* B(ShortestStrikeBundle(x, y), 1);
guard $\text{New}(r) \wedge \text{StrictAttack}(r, t) \wedge l < n$	$\text{B}(t, l) \rightarrow \text{a}(r, l) * \text{B}(r, l+1) * \text{B}(t, l)$
guard $\text{New}(r) \wedge \text{StrictAttack}(r, t) \wedge l = n$	$\text{B}(t, l) \rightarrow \text{a}(r, l) * \text{B}(t, l)$
guard $\forall r \in \mathbf{L}_{\text{stb}}^{\text{H}}(p) \bullet \neg(\text{New}(r) \wedge \text{StrictAttack}(r, t))$ **od**	$\text{B}(t, l) \rightarrow \Lambda$

Appendix B. Wide Illustrations


Campaign Board
Sub-boards
Operational Theater Board
Cells
Tactical Control Element (TCE) Board
• • •
• • •

Strategic Level Board:

- Sub-boards: operational theaters
- Cells: TCE or results of combining several TCEs

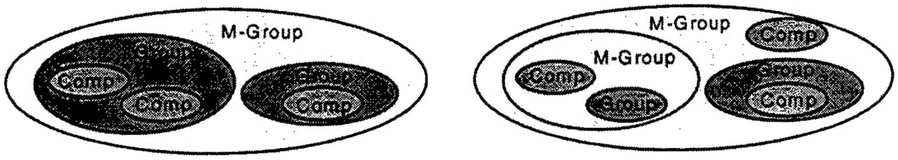
Operational Level Board:

- Sub-boards: TCEs
- Cells: coarse grain

Tactical Level Board:

- Cells: fine grain

Figure 15. Strategic-Operational-Tactical Game Hierarchy



(a) Homogeneous Hierarchy

(b) Heterogeneous Hierarchy

Figure 16. A hierarchy of mega-groups

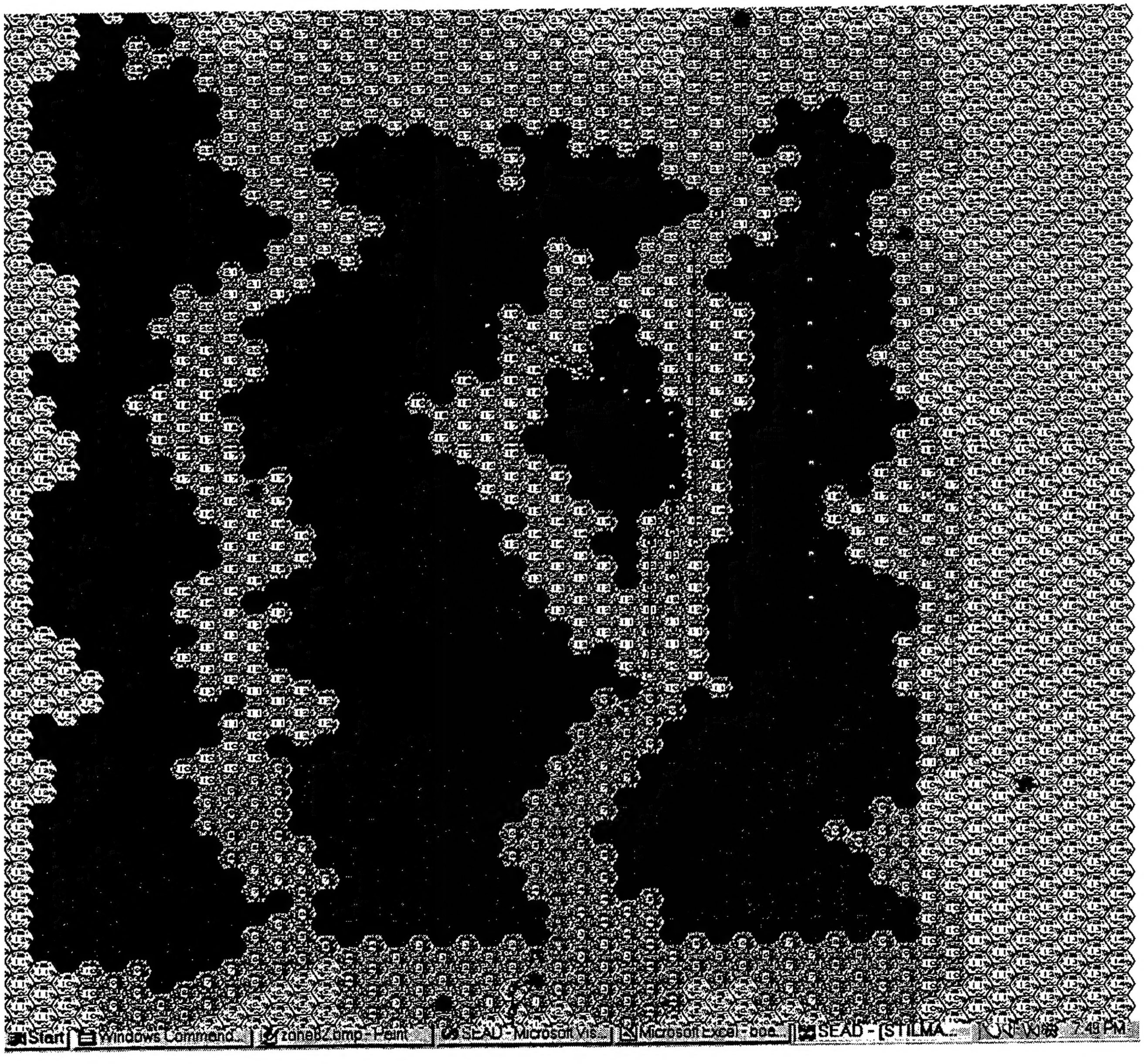


Figure 17. An LG attack Zone